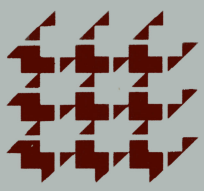


DIMACS

Series in Discrete Mathematics and Theoretical Computer Science

Volume 37

Mathematical Hierarchies and Biology

DIMACS Workshop
November 13–15, 1996

Boris Mirkin
F. R. McMorris
Fred S. Roberts
Andrey Rzhetsky
Editors

American Mathematical Society

Selected Titles in This Series

37 **Boris Mirkin, F. R. McMorris, Fred S. Roberts, and Andrey Rzhetsky, Editors,** Mathematical Hierarchies and Biology

36 **Joseph G. Rosenstein, Deborah S. Franzblau, and Fred S. Roberts, Editors,** Discrete Mathematics in the Schools

35 **Dingzhu Du, Jun Gu, and Panos M. Pardalos, Editors,** Satisfiability Problem: Theory and Applications

34 **Nathaniel Dean, Editor,** African Americans in Mathematics

33 **Ravi B. Boppana and James F. Lynch, Editors,** Logic and random structures

32 **Jean-Charles Grégoire, Gerard J. Holzmann, and Doron A. Peled, Editors,** The SPIN verification system

31 **Neil Immerman and Phokion G. Kolaitis, Editors,** Descriptive complexity and finite models

30 **Sandeep N. Bhatt, Editor,** Parallel Algorithms: Third DIMACS Implementation Challenge

29 **Doron A. Peled, Vaughan R. Pratt, and Gerard J. Holzmann, Editors,** Partial Order Methods in Verification

28 **Larry Finkelstein and William M. Kantor, Editors,** Groups and Computation II

27 **Richard J. Lipton and Eric B. Baum, Editors,** DNA Based Computers

26 **David S. Johnson and Michael A. Trick, Editors,** Cliques, Coloring, and Satisfiability: Second DIMACS Implementation Challenge

25 **Gilbert Baumslag, David Epstein, Robert Gilman, Hamish Short, and Charles Sims, Editors,** Geometric and Computational Perspectives on Infinite Groups

24 **Louis J. Billera, Curtis Greene, Rodica Simion, and Richard P. Stanley, Editors,** Formal Power Series and Algebraic Combinatorics/Séries formelles et combinatoire algébrique, 1994

23 **Panos M. Pardalos, David I. Shalloway, and Guoliang Xue, Editors,** Global Minimization of Nonconvex Energy Functions: Molecular Conformation and Protein Folding

22 **Panos M. Pardalos, Mauricio G. C. Resende, and K. G. Ramakrishnan, Editors,** Parallel Processing of Discrete Optimization Problems

21 **D. Frank Hsu, Arnold L. Rosenberg, and Dominique Sotteau, Editors,** Interconnection Networks and Mapping and Scheduling Parallel Computations

20 **William Cook, László Lovász, and Paul Seymour, Editors,** Combinatorial Optimization

19 **Ingemar J. Cox, Pierre Hansen, and Bela Julesz, Editors,** Partitioning Data Sets

18 **Guy E. Blelloch, K. Mani Chandy, and Suresh Jagannathan, Editors,** Specification of Parallel Algorithms

17 **Eric Sven Ristad, Editor,** Language Computations

16 **Panos M. Pardalos and Henry Wolkowicz, Editors,** Quadratic Assignment and Related Problems

15 **Nathaniel Dean and Gregory E. Shannon, Editors,** Computational Support for Discrete Mathematics

14 **Robert Calderbank, G. David Forney, Jr., and Nader Moayeri, Editors,** Coding and Quantization: DIMACS/IEEE Workshop

13 **Jin-Yi Cai, Editor,** Advances in Computational Complexity Theory

12 **David S. Johnson and Catherine C. McGeoch, Editors,** Network Flows and Matching: First DIMACS Implementation Challenge

11 **Larry Finkelstein and William M. Kantor, Editors,** Groups and Computation

10 **Joel Friedman, Editor,** Expanding Graphs

9 **William T. Trotter, Editor,** Planar Graphs

8 **Simon Gindikin, Editor,** Mathematical Methods of Analysis of Biopolymer Sequences

7 **Lyle A. McGeoch and Daniel D. Sleator, Editors,** On-Line Algorithms

(*Continued in the back of this publication*)

This page intentionally left blank

https://doi.org/10.1090/dimacs/037

DIMACS

Series in Discrete Mathematics
and Theoretical Computer Science

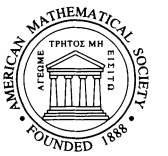
Volume 37

Mathematical Hierarchies and Biology

DIMACS Workshop
November 13–15, 1996

Boris Mirkin
F. R. McMorris
Fred S. Roberts
Andrey Rzhetsky
Editors

NSF Science and Technology Center
in Discrete Mathematics and Theoretical Computer Science
A consortium of Rutgers University, Princeton University,
AT&T Labs, Bell Labs, and Bellcore



American Mathematical Society

This DIMACS volume contains a set of refereed papers written mostly by participants in the workshop "Mathematical Hierarchies and Biology" held at DIMACS on November 13–15, 1996. The papers provide a contemporary sample of many new results in hierarchy theory with applications in biology, psychology, data analysis, and systems engineering and should be of use to researchers in discrete mathematics, computational biology, and many other areas.

1991 *Mathematics Subject Classification.* Primary 92Bxx; Secondary 05Cxx.

Library of Congress Cataloging-in-Publication Data

Mathematical hierarchies and biology : DIMACS workshop, November 13–15, 1996 / Boris Mirkin... [et al.], editors.

p. cm. — (DIMACS series in discrete mathematics and theoretical computer science, ISSN 1052-1798 ; v. 37)

"NSF Science and Technology Center in Discrete Mathematics and Theoretical Computer Science, a consortium of Rutgers University, Princeton University, AT&T Labs, Bell Labs, and Bellcore."

Includes bibliographical references and index.

ISBN 0-8218-0762-5 (hc : alk. paper)

1. Graph theory—Congresses. 2. Hierarchies—Congresses. 3. Biomathematics—Congresses. I. Mirkin, B. G. (Boris Grigor′evich) II. DIMACS (Group) III. NSF Science and Technology Center in Discrete Mathematics and Theoretical Computer Science. IV. Series.

QA166.M365 1997

511′.5—dc21 97-26706

CIP

Contents

Foreword

The Workshop on "Mathematical Hierarchies and Biology", held in November 1996, was part of DIMACS Special Focus on Mathematical Support for Molecular Biology that began in 1994. We would like to express our appreciation to Boris Mirkin as chair of the committee and to Buck McMorris, Fred Roberts, and Andrey Rzhetsky assisting him for their efforts to organize and plan this successful workshop as well as for editing this volume of papers.

The special year encouraged collaborations among very different research communities, and this volume records one of many workshops in which this was achieved. We also extend our thanks to Fred Roberts and Joachim Messing as chairs of the special year committee, Lawrence Shepp and Michael Waternam as co-chairs, Martin Farach as assistant chair, and Sampath Kannan as publicity chair for their efforts to coordinate the broad range of activities.

DIMACS gratefully acknowledges the generous support that makes these programs possible. The National Science Foundation, through its Science and Technology Center program, the New Jersey Commission on Science and Technology, DIMACS partners at Rutgers, Princeton, AT&T Labs, Bell Labs, and Bellcore generously supported the special year, and we thank NSF for a special supplementary award that helped to support the workshop that led to this volume. Additional support for the special year was obtained from BIOSYM Tech, the Centers for Disease Control, Ciba-Geigy, Merck Research, the National Center for Human Genome Research, the National Institute for Allergy and Infectious Diseases, the National Security Agency, Roche Molecular Systems, Sandia, and SmithKline Beecham.

Fred S. Roberts
Director

Bernard Chazelle
Co-Director for Princeton

Stephen R. Mahaney
Associate Director for Research

This page intentionally left blank

Preface

Viewing hierarchies as combinatorial objects underlies many important areas of current scientific investigation. Examples of areas where this is the case are biology, evolutionary studies and taxonomy, general classification theory, data/knowledge bases, cognitive models and linguistical structures, industrial control, etc. The mathematical approach to the study of hierarchies represents the theoretical underpinning of many important areas of current scientific investigation. Biology has benefited from this research and has also stimulated the mathematical study of hierarchies.

This collection presents a set of refereed papers written mostly by the participants in the Workshop "Mathematical Hierarchies and Biology" held at DIMACS 13-15 November 1996. The papers and some of the results described implement many of the discussions and suggestions for revision made by the other participants and editors. Of two extreme formats of a scientific publication, monographs and journals, the current edition has taken some features of both, combining review papers and papers presenting the latest results.

The papers can roughly be organized into the following four areas:

1. Combinatorial modeling of the evolutionary processes (G. Estabrook; C. Nehaniv and J. Rhodes; M. Bonet, C. Phillips, T. Warnow, and S. Yooseph; R. Page and M. Charlestone; O. Eulenstein, B. Mirkin, and M. Vingron; A. Rzhetsky, F. Ayala, L. Hsu, C. Chang, and A. Yoshida; M. Steel, V. Moulton, and C. Tuffley)
2. Reconstructing trees from dissimilarity data (K. Atteson; O. Gascuel; A. Guénoche; B. Leclerc and V. Makarenkov; F.-J. Lapointe and P.-A. Landry; the paper by M. Steel et al in item 1 can be put here, too)
3. Related mathematical issues (P. Pardalos and X. Deng; P. Hansen and D. Werra; F. Roberts and L. Sheng; E. Kubicka, G. Kubicki, and F. McMorris; T. Przytycka; F. McMorris and R. Powers)
4. Clustering and data analysis (A. Dress; I. Van Mechelen, S. Rosenberg, and P. De Boeck; L. Hubert, P. Arabie, and J. Meulman; D. Carrol and G. De Soete; B. Mirkin; A. Meystel)

The reader should be aware that there are many other aspects of mathematical hierarchies covered and many interconnections among the papers beyond the grouping according to the four classes above. The subjects can be identified using the contents and index to the volume. The interconnections can be found based on mathematical problems analyzed, formalisms used, and on the substantive problems involved.

The papers in the volume provide a contemporary sample of many new results in hierarchy theory with applications in biology, psychology, data analysis and

systems engineering, and should be of use to researchers in discrete mathematics, computational biology and many other areas.

B. Mirkin
F.R. McMorris
F.S. Roberts
A. Rzhetsky

https://doi.org/10.1090/dimacs/037/01

DIMACS Series in Discrete Mathematics
and Theoretical Computer Science
Volume **37**, 1997

Ancestor-Descendant Relations and Incompatible Data: Motivation for Research in Discrete Mathematics

George Estabrook

ABSTRACT. The species in a collection of biological species may have evolved in such a way that each species has a unique species as its immediate ancestor and that the entire collection has a single most recent common ancestor. Some or all of the species ancestral to those in the collection may be missing from the collection. The concept "is an ancestor of" can be construed as a relation on the collection of species extended to include any missing ancestors, the Ancestor relation.

Comparative biological data for such a collection are partitions of the collection into classes considered equivalent with respect to some basis for comparison. A task of interest to comparative biologists is to reconstruct the unknown Ancestor relation from comparative data. In the mid 1960's some well known biologists, including Wilson (1965), Camin and Sokal (1965), and Hennig (1966) began to realize that often comparative data were somehow contradictory, so that this reconstruction was confounded.

Estabrook (1968) formulated this problem in the terms of discrete mathematics. A partition accurately reflects the Ancestor relation if its classes are all convex on the Hasse diagram of the Ancestor relation. Two partitions are contradictory if there does not exists any ancestor-like relation on whose Hasse diagram all classes of both are convex. A desire to understand these contradictions, and to find realistic ways to resolve them, have motivated related research in discrete mathematics since that time. This essay reviews some progress and describes some of the present open questions.

1. Biological Concepts

For the past 140 years, students of natural history have been interested to guess how species are related historically through the process of evolutionary change, first clearly articulated to western science by Wallace and Darwin in 1858. Increasing during the twentieth century, natural historians have stated their guesses with tree-like diagrams. These are often somewhat vague, such as the cactus diagram of Bessey (1915), the splattered rain drops of Benson (1957), or the fans of Cronquist (1968); but also sometimes more specific, such as the fairly well resolved branching tree of flowering plant orders of Takhtjan (1969) or the quite explicit diagrams of estimated Ancestor relations of Hall and Clements (1923), who indicated right on the diagram where specific features are hypothesized to have undergone evolutionary change.

The contemporary concepts of phylogenetic tree are diverse, and controversial. Some remain vague. The concept underlying this discussion is illustrated in Figure 1. On the left is a diagram of branching lines (the phyletic lines), with one line at the

1991 *Mathematics Subject Classification.* Primary05C05, 05C90; Secondary 06A12.

bottom that successively branches to produce several line ends near the top. The vertical direction up the page represents time passing from past at the bottom to more present nearer the top. The time scale is not specified beyond monotone. The horizontal direction means nothing beyond providing the space to draw the branching of the lines. Near the upper ends of the lines often occur symbolic or real names of species or other grouping of "related" organisms. In this discussion these groupings will always be called species, both to simplify exposition and to avoid complications not so central to the concepts to be presented.

The concept of species is also diverse, controversial, and sometimes vague. Here, two organisms are considered to "belong to the same species" if they would be capable of breeding together to create reproductively competent progeny, provided they were of compatible mating types, reproductively ready, and close enough in time and space to effect sexual union of gametes. Many natural historians believe that "belong to the same species" is almost an equivalence relation whose "almost" classes are species. This biological concept is useful as an idea, even if practical problems of testing species membership make it impossible to actually check most organisms. Generally creatures that look very similar except in ways that vary among known families or that vary as plastic developmental responses to different environments, are considered to belong to the same species. One important reason why this "almost" equivalence is not an equivalence arises with the origin of a new species from its immediate ancestor. A speciation event occurs where rapid evolutionary changes result in a descendant species with some features different from its immediate ancestor. It seems reasonable that progeny should belong to the same species as their parents, yet from ancestral species to descendant species there exists an unbroken chain of parents and their children. If every individual belongs to some species then at some point a parent must belong to one species and its progeny to another.

Different resolutions of this apparent problem contribute to the rise of contending schools of thought among evolutionary and systematic biologists. One resolution suggests that the time scale on which we view ancestor-descendant relations among species is very large compared with the generation time of an individual, making the few hundreds or thousands of generations during which evolutionary change processes take place seem essentially invisible. Individuals that are involved in the rapid

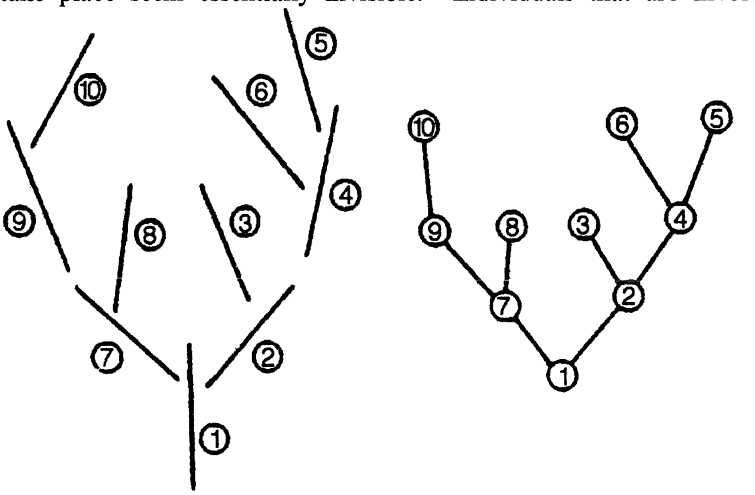


Figure 1.

evolutionary change process, and thus belong ambiguously to the ancestor or the descendant species, are considered to belong to neither. We erase these from the phylogenetic tree, as shown by the small gaps in Figure 1. The resulting unbroken line segments each represent a single species, of which there are 10. Some species are ancestors of others, expressed as the Ancestor relation, i.e., pairs (a,b) of species in which a is an ancestor of b. The Ancestor relation is reflexive, antisymmetric, and transitive, and satisfies treeness: for any two species x and y, the descendants of x and the descendants of y are either disjoint or one set of descendants contains the other. The Hasse diagram for the Ancestor relation is also shown on the right.

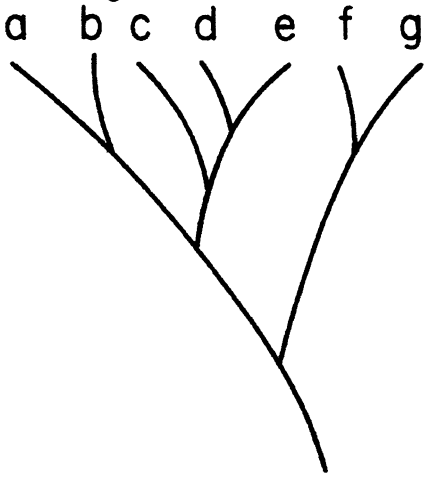


Figure 2.

The phylogenetic tree of Figure 2 is shown in Figures 3 and 4, where arrows point to speciation events at the beginning of a new branch. The resulting ancestor relation is shown to the right. Different speciation events result in different ancestor relations, even though the phylogenetic trees are the same. Figure 5 illustrates an extinct species, X, which gave rise to b at evolutionary event *. Unobserved extinct species will be shown as empty circles in Hasse diagrams that include them. These diagrams look different, yet they are related by being derived from the same phylogenetic tree.

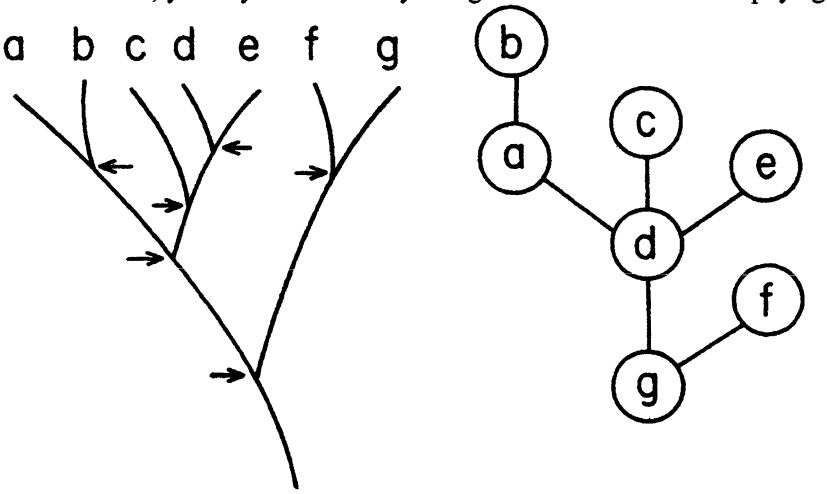


Figure 3.

Natural questions arise. Can any Ancestor relation arise from a given phylogenetic tree? (No) Are there enlightening characterizations of the Ancestor relations determined by a given phylogenetic tree? of phylogenetic trees determined by a given Ancestor relation? Some progress has been made, but I believe there is still room for an elegant treatment.

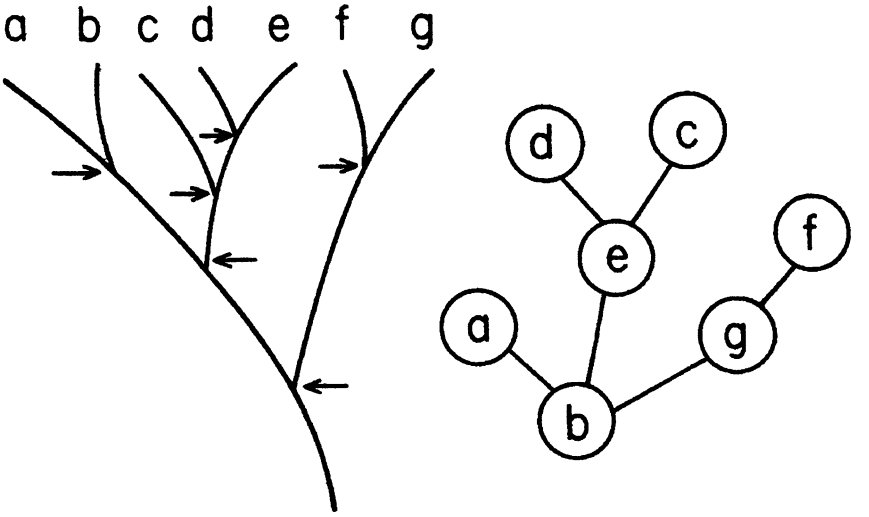


Figure 4.

Commonly, evolutionary biologists have comparative data with which to guess about the evolutionary history of a study group of related species. One aspect of this history that is interesting to guess is the historically *true* Ancestor relation, denoted throughout with a capital letter. Relations that are ancestor-like serving as estimates or possibilities but not presumed to be true are referred to as ancestor relations. Comparative data are illustrated in Figure 6, where species of flowers: *a*, *b*, *c*, *d*, *e*, are compared with respect to 4 different bases for comparison: flower shape, leaf petiole,

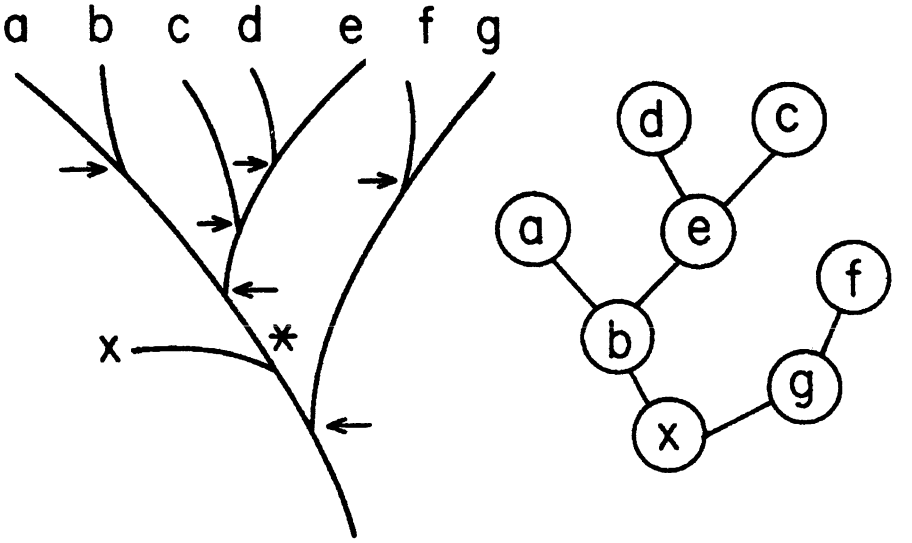


Figure 5.

phyllotaxy and hair. Each basis for comparison provides a criterion to decide whether two species are the same or different with respect to that basis. Such a basis is an equivalence, called a character by comparative biologists, whose equivalence classes are called character states. In Figure 6, the states of each character have been labeled with A and B arbitrarily to distinguish them. State membership of each species for each character is shown in Figure 6 with a matrix. To make a relationship between such comparative data and the evolutionary history for these species, we hypothesize that the change in feature that distinguishes one character state from another arose during a speciation event at some particular place on the phylogenetic tree (Estabrook et al. 1975).

To understand how we might use comparative data to guess about evolutionary history we examine how character states would arise at speciation events on a phylogenetic tree. Figure 7 shows where the features of a single basis for comparison changed in the phylogenetic tree of Figure 2. For example, the species may be some flowering plants, and the basis for comparison may be the color of the flower. The ancestral flower color may have been yellow but changed to blue during the speciation at the lower arrow and subsequently to white during the speciation at the upper arrow. All the other speciations in the evolutionary history of these species involved changes in features other than flower color. This results in three character states for flower color: the yellow species represented by f and g among modern species; the blue species represented by *c*, *d* and *e* among modern species; and the white species represented by modern species *a* and *b*. These character states may also contain unobserved extinct species as well. If we collapse the phylogenetic tree into the states of the character flower color, these states inherit a tree order, whose Hasse diagram is

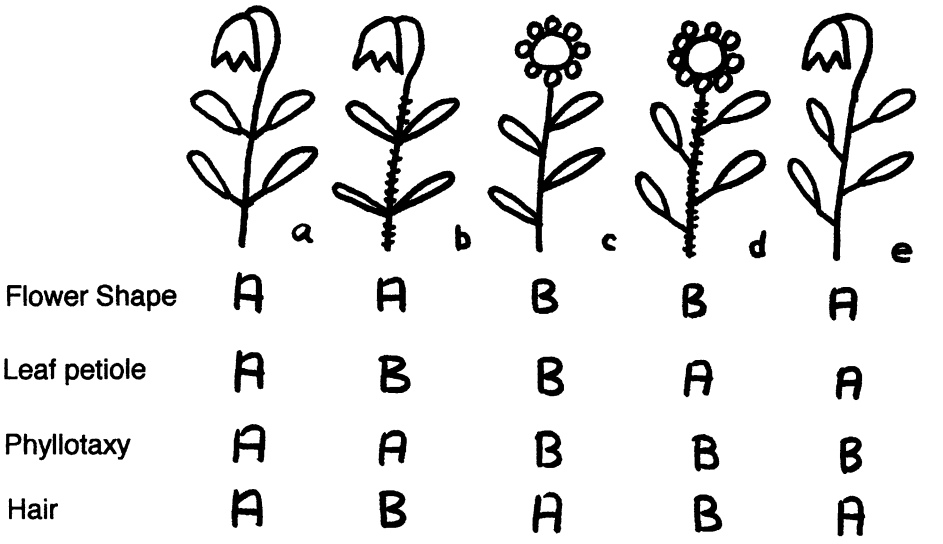


Figure 6.

also shown in Figure 7, called the character state tree of flower color. It is possible that some of the unobserved extinct ancestors belonged to character states whose defining features are not represented among the observable modern species under study. Figure 8 shows what would become of character state tree i if there were an additional change in flower color at speciations indicated with ii or iv. A discussion of character state trees and phylogenetic trees is found in Estabrook (1984). Not all

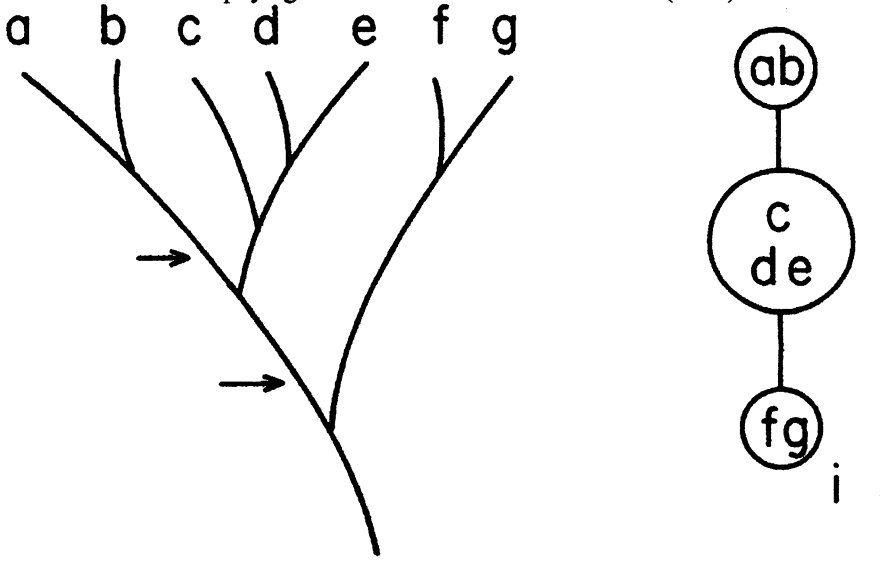


Figure 7.

character state trees can be the result of collapsing the Ancestor relation, or the phylogenetic tree from which it is derived, as described above, but only those that correctly indicate true speciation events, which are called true character state trees. A character state tree is *true* (as a hypothesis about the Ancestor relation) if it is a (lower semi-lattice) homomorphic image of the Hasse diagram of the Ancestor relation (Estabrook et al. 1975).

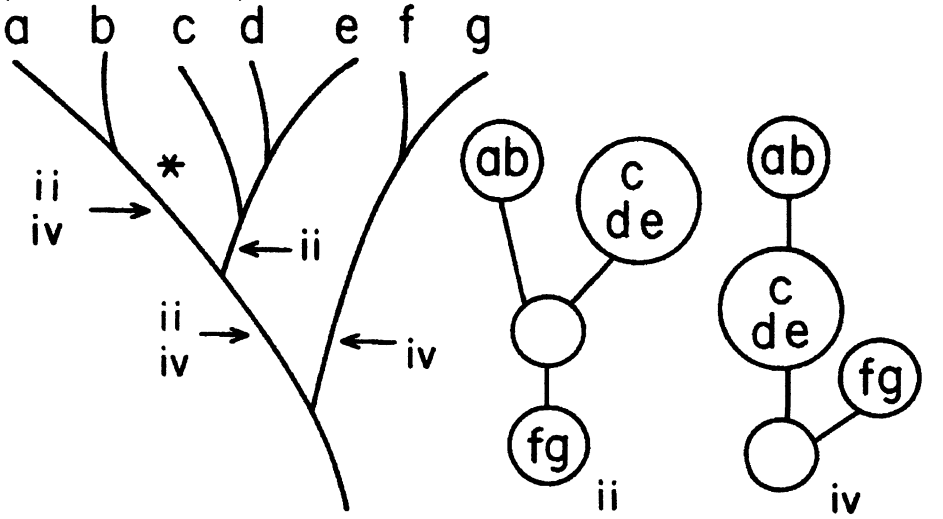


Figure 8.

The character state trees for two different characters can be "added" to get the character state tree that is their *sum*, by taking the speciation events creating distinctions for either character to create the distinctions in the sum character, as shown in Figure 9. Here we use the phylogenetic tree to see what is the sum but the terms alone determine the sum. Just as there are many phylogenetic trees that can produce a given ancestor relation, so there are many that can produce a given

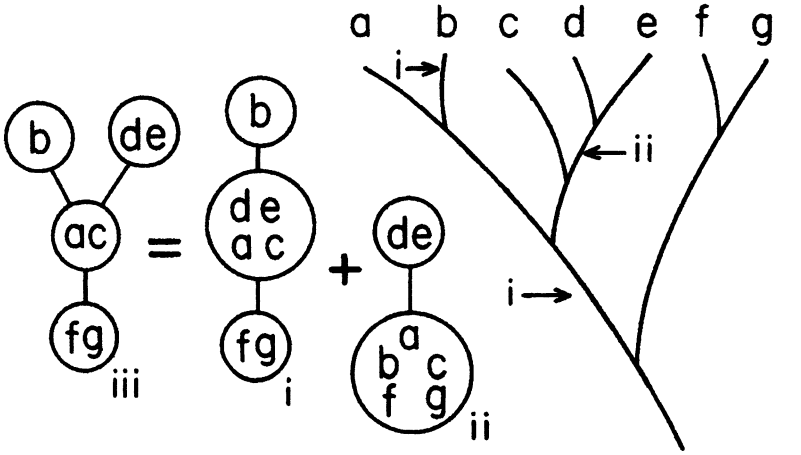


Figure 9.

character state tree. The phylogenetic trees that can produce the sum is the intersection of those that can produce the terms (Estabrook, 1972). A technique to determine the sum without reference to a specific phylogenetic tree was formulated by F. R. McMorris, reported by Estabrook et al. (1976), and is shown in Figure 10. The technique consists of representing each character state as the subset of the collection of species under study comprised of the union of all character states descendant from it,

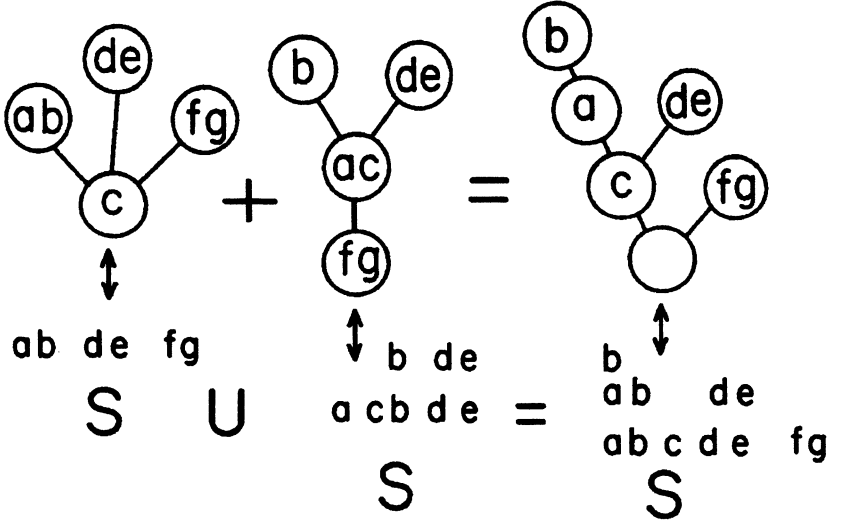


Figure 10.

including itself. All such subsets for a character state tree is a *tree of subsets*: either these subsets are disjoint or they contain each other. In Figure 10 the double arrows indicate the bi-unique correspondence between trees of subsets and character state trees. The sum of two character state trees is the character state tree of the union of their trees of subsets.

Not every pair of character state trees can be added; the union of the trees of subsets of two character state trees may not be a tree of subsets, as illustrated in Figure 11. The problem will always be a violation of treeness. If two character state trees cannot be added, then their sets of associated phylogenetic trees are disjoint. Consider again comparative data. Evolutionary biologists have hypothesized that the distinctions used to define characters correspond to speciation events on the phylogenetic tree of the species under study and have expressed these hypotheses as character state trees. When character state trees cannot be added it is because there is no structurally possible phylogenetic tree that can be collapsed into each of them by placement of hypothetical speciation events. Such character state trees are logically incompatible (mutually contradictory) as hypotheses about the Ancestor relation. At least one of them must be false.

Compatible character state trees can be added to get a new character state tree that is a refinement of each of its terms; the sum approximates some ancestor relation (perhaps not THE Ancestor relation) in more detail. Note that possible ancestor relations are character state trees with at most one species in any state. So the collection of all possible character state trees includes the Ancestor relation we are trying to estimate. If we limit unobserved extinct ancestors to those hypothesized to be immediate ancestors of more than one species, then for a finite study collection S of

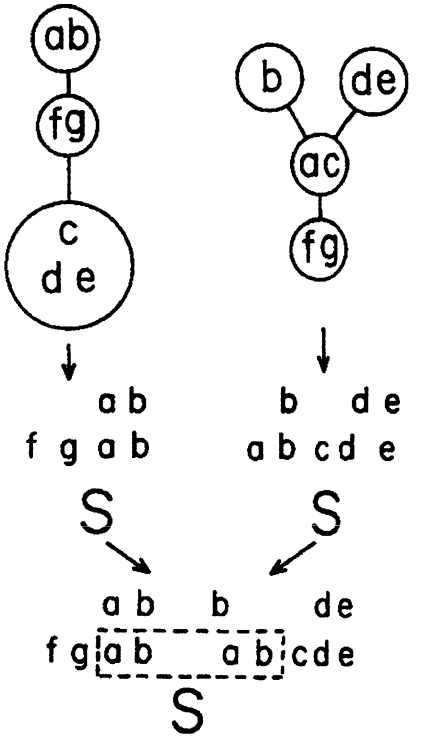


Figure 11.

species, only a finite number of possible character state trees exists. They have been counted by McMorris and Zaslavsky (1981). When S has three species they number 32, and are shown in Figure 12. The relation "can be refined into" (in the above sense) induces a lower semi lattice structure on character state trees, where upper bounds exist only for compatible character state trees but lower bounds always exist. The Hasse diagram for the character state trees of Figure 12 are shown in Figure 13. Identification or creation of results to describe more completely the nature of this family of semilatices would be interesting.

Compatibility does not guarantee truth, but incompatibility does guarantee that at least one character state tree is false, and provides the evolutionary biologist with an objective basis for evaluating data. Suppose many character state trees have been found to be pairwise compatible. Then there is a single phylogenetic tree (and

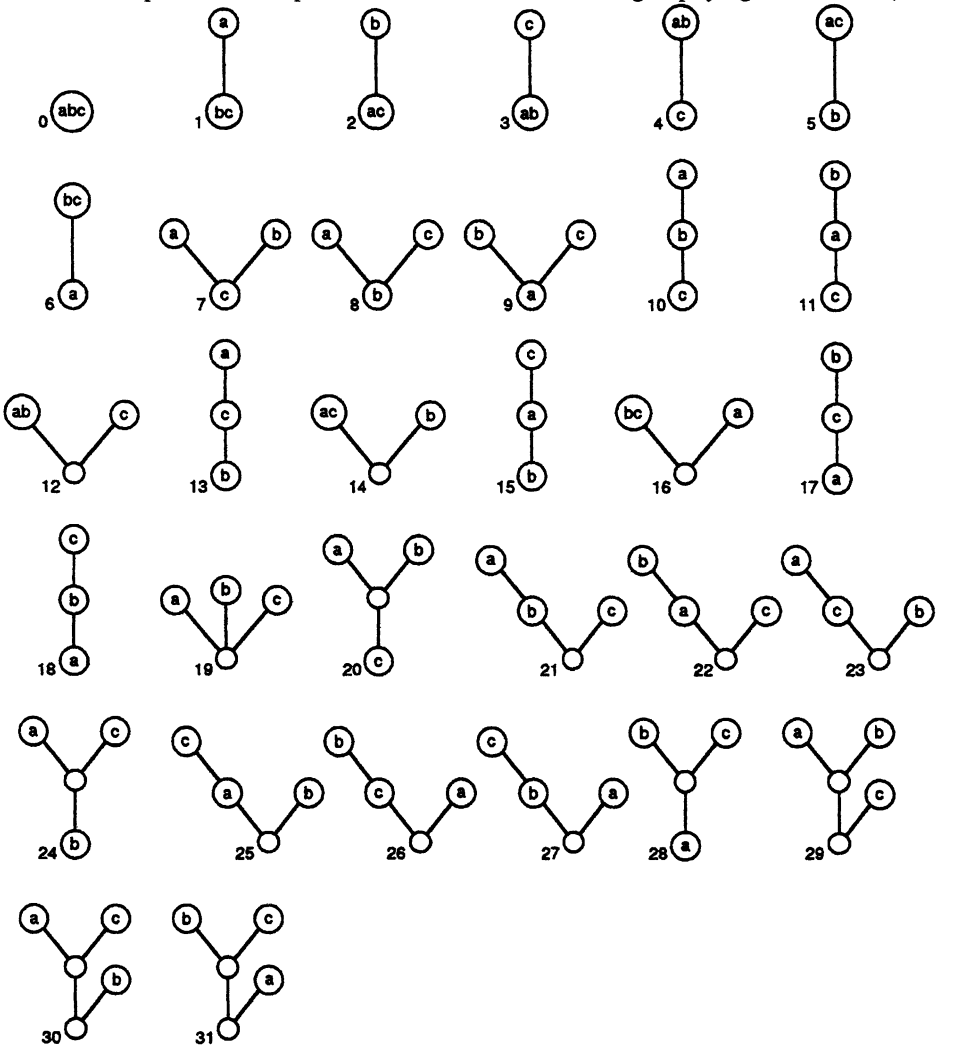


Figure 12.

speciation events) that will accommodate them all at once (Estabrook et al. 1976a, 1976b). This result, illustrated in Figure 14, guaranteed the accuracy of extant algorithms to analyze comparative data and enable discrete mathematics to provide a useful service to evolutionary biologists.

Camin and Sokal (1965), using fossil horse bone data, were among the first to explicitly formulate character state trees as hypotheses of evolutionary relationships. It was clear to them that conflicts among these character state trees would have to be resolved somehow to make an estimate of the Ancestor relation. Wilson (1965), Hennig (1966) and LeQuesne (1969) all knew that violations of treeness evidence conflicting hypotheses and all three described mechanically distinct but logically equivalent tests for the compatibility of characters with two states. Camin and Sokal (1965) suggested a minimizing criterion, studied by Estabrook (1968). Criteria of this

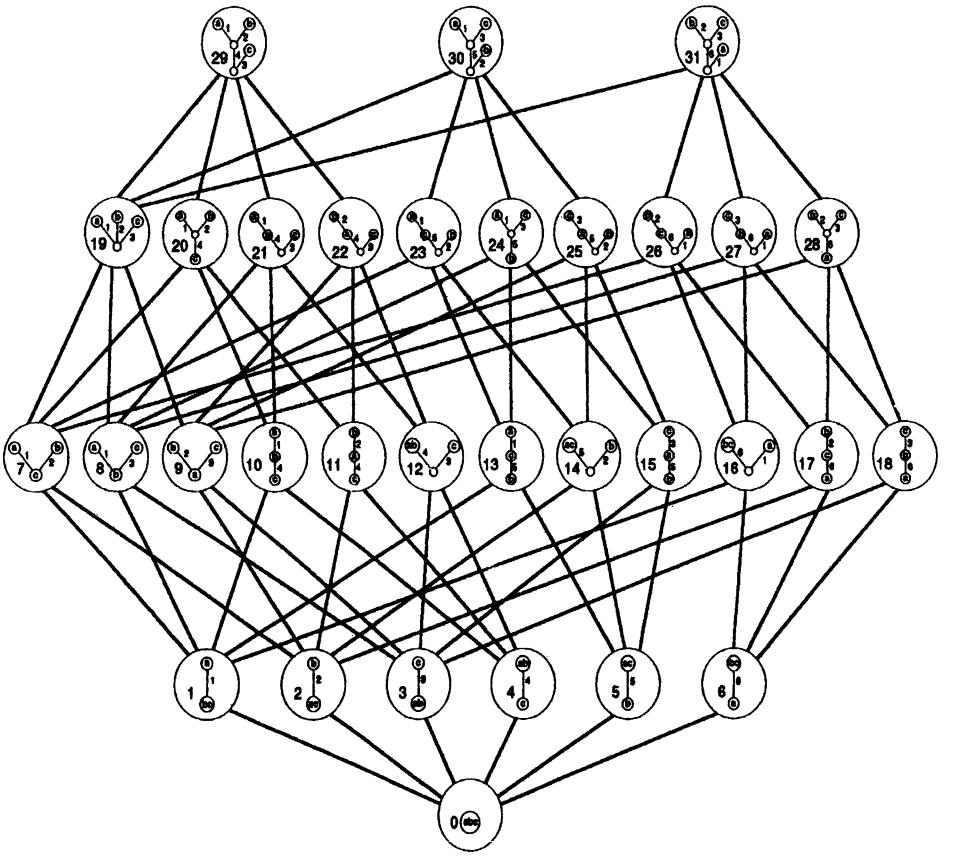


Figure 13.

kind have been devised for three decades, and heuristic parsimony algorithms for them implemented in a variety of commercially available computer programs, such as PAUP (Swofford 1991). But a better understanding of the causes of incompatibility was needed so to more credibly resolve them.

2. Causes of Incompatibility

If character state trees (CSTs) are incompatible, then at least one is false. One way for a CST to be false is for the stated direction of evolutionary change to be false. If this were so, then some other state could be chosen for ancestral and the CST directed away from it to produce a true CST. McMorris (1977) showed that in the case of a pair of incompatible CST's with two states directed so that the one with the most species in it was taken as ancestral, they would always remain incompatible if

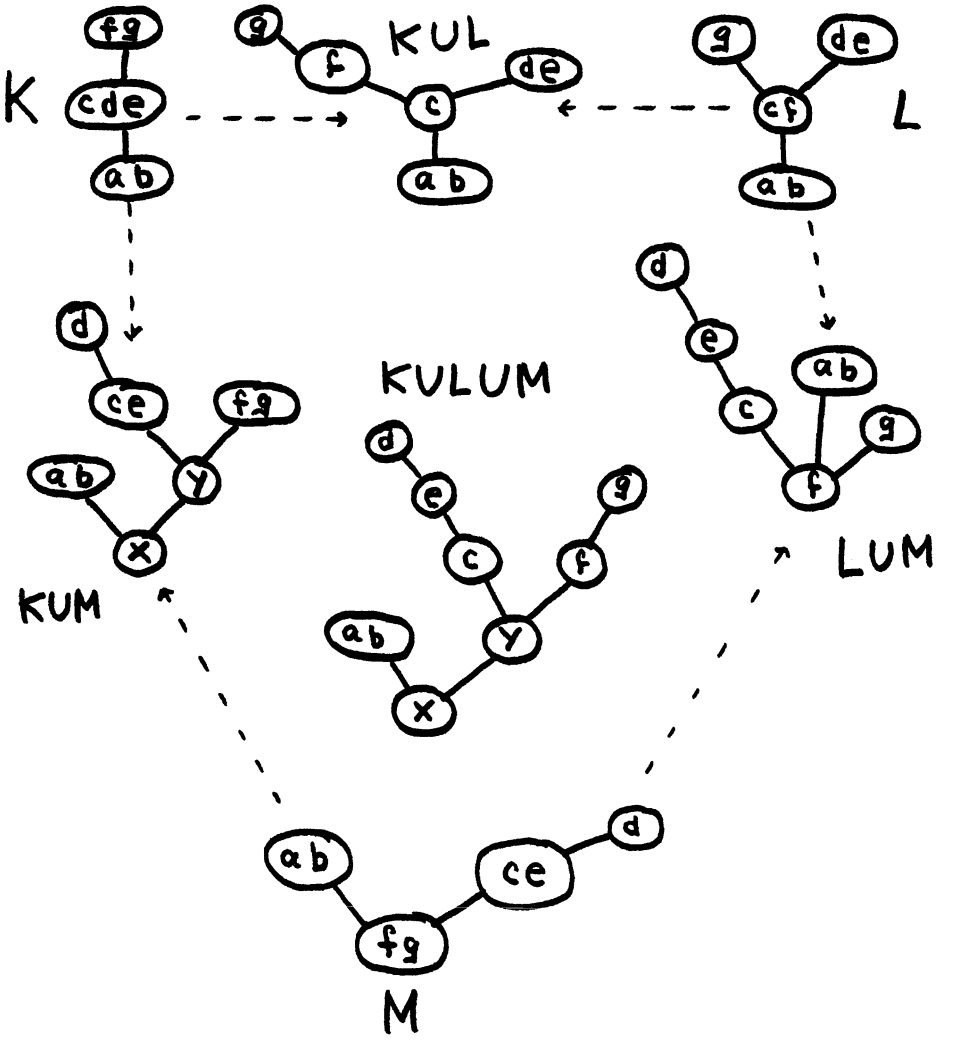


Figure 14.

either or both CST were directed the other way. A similar result for CSTs with any number of states was presented by Estabrook and Meacham (1979), who defined one CST to be *undirected tree equivalent* (ut) to another if the edges of one could be redirected to convert it to the other, as illustrated by the four CSTs of Fig 15. Here "I ut II" is false because the states are different; and "I ut III" is false because the undirected edges are different; but "I ut IV" is true because the lower edge of I can be reversed to create IV. A CST is directed *common equal primitive* (cep) if for every directed edge the sum of the species in the states toward which it is directed does not exceed half the total number of observed species. CST I of Fig 15 is not cep because the lower arrow points towards states containing in sum four species, which exceeds 7/2; CST IV is cep.

Every ut equivalence class contains a cep member. If two cep CSTs are incompatible, then so will be any pair of CSTs chosen from their respective ut equivalence classes. Contrapositively, if any pair of CSTs is compatible so will be any pair of cep CSTs from their respective ut classes. The sum of compatible cep CSTs is cep. Thus a compatibility analysis of cep representatives of data derived CSTs using the results of Estabrook et al (1976a&b) reveals only incompatibilities that are not the result of conflicting directions. Meacham (1984) discusses the ramifications of this for comparative biology. The enumeration of cep CST's and their refinement lower semi-lattice deserves study.

Suppose CST I of Fig 15 were true. CST III is incompatible with CST I (and thus false) not just because the direction of the hypothesized evolutionary change between state $\{f,g\}$ and state $\{d,e\}$ is wrong, but also because the state hypothesized to be derived from $\{a,b,c\}$ is also wrong, an error in proximity's. Without proximity's

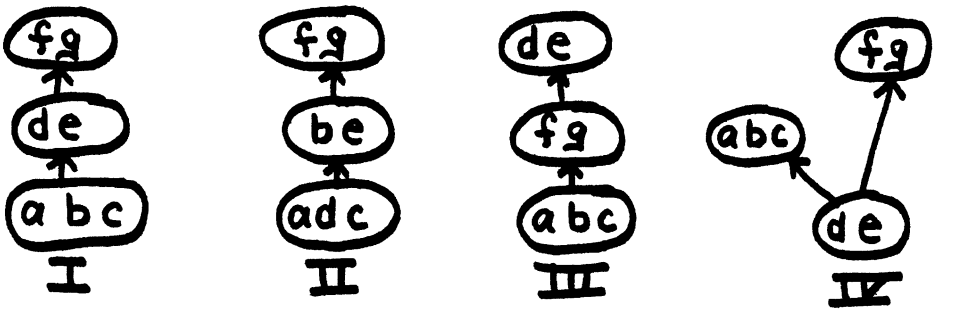


Figure 15.

specified, data take the form of *qualitative taxonomic characters* (QTCs) whose character states are the equivalence classes of a basis for comparison among species. Two QTCs are *potentially compatible* if there exists a way to connect the states of each with directed edges to make compatible CSTs. If two CSTs are ut incompatible but potentially compatible as QTCs their incompatibility is a disagreement in proximity's, which could be resolved by hypothesizing different proximity's. These concepts are illustrated in Fig 16.

Fitch (1975) and Estabrook and Landrum (1975) conjectured logically equivalent n/c conditions for two QTCs to be potentially compatible. The procedure is illustrated in Fig 17. with a data matrix in which capital letters name the states of QTCs I, II, and III, labeling the rows, to which belong species, labeling the columns with lower case letters. For each pair of QTCs, their state contingency matrix contains an entry (X) if

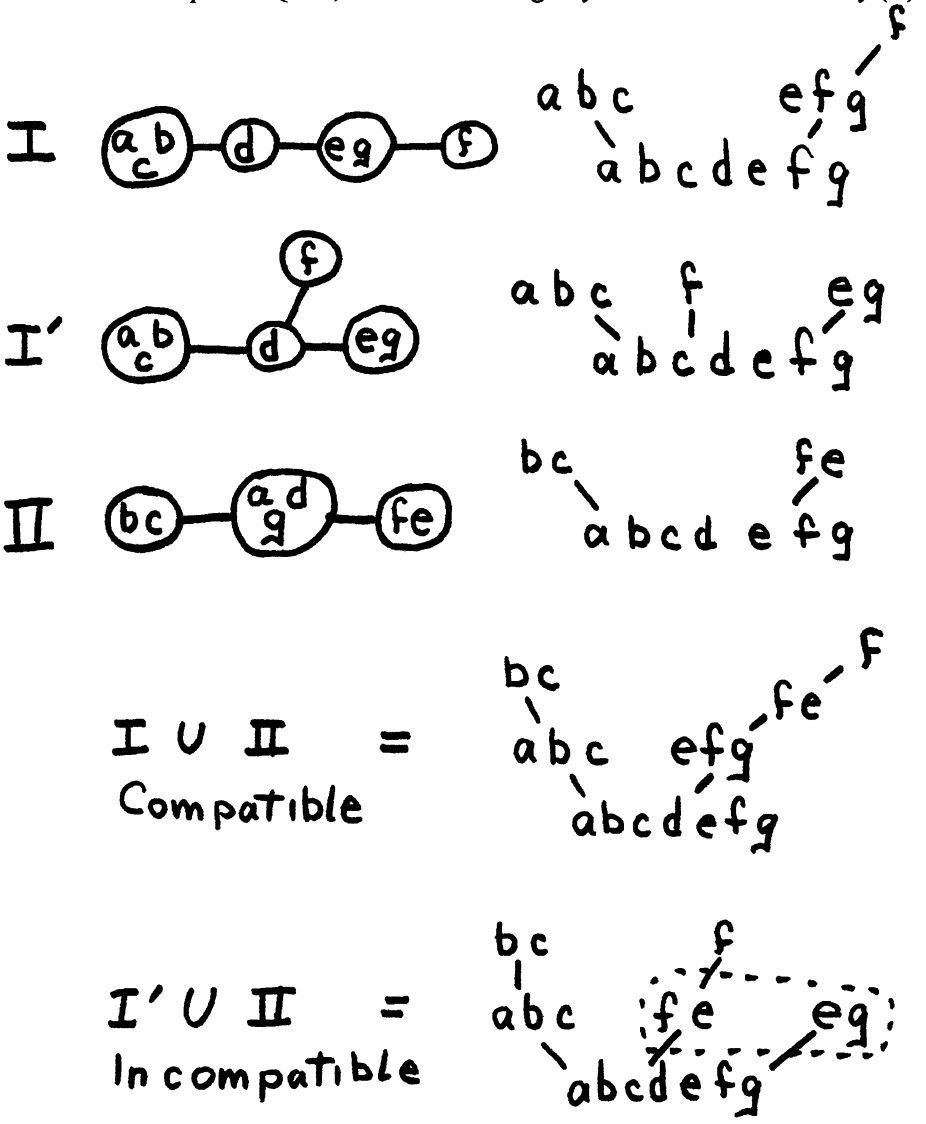


Figure 16.

some species occurs in both the row and the column state. A closed path of rook's moves, absent in the left matrix but shown in the right, indicated incompatibility. This conjecture, together with an algorithm for implementing it, was proved by Estabrook and McMoris (1977) and applied by Boulter et al (1979) to amino acid sequence data. Fitch (1977) showed by example that pairwise potential compatibility of all pairs of QTCs in a set was not sufficient for their potential set-wise compatibility, as shown in Figure 18a and 18b. Here the proximity's for I and II to realize their potential compatibility require state B in the middle, but the proximity's for II and III to realize their potential compatibility require state A to be in the middle. Since states A and B cannot both be in the middle, I, II, and III cannot simultaneously be compatible.

For two state QTCs, there is only one undirected tree, so incompatibility of

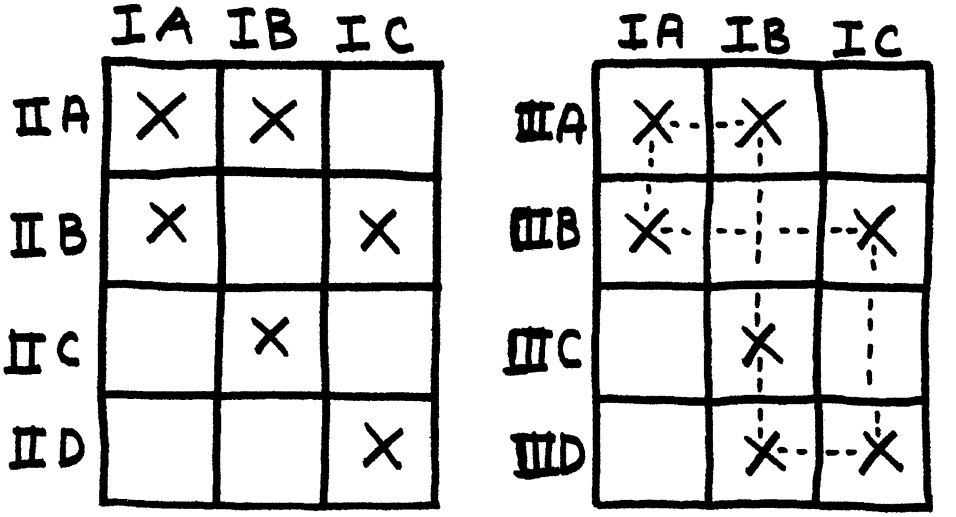


Figure 17.

proximity can never exist, and the methods of Estabrook et al (1976b) and McMorris (1977) described above provide the basis for fast and efficient analysis of data. Because they address only this case of two-state QTCs, the results of Agarwala et al (1995), although they may provide a basis for speeding up already fast algorithms, seem not to address this difficult problem. Gavril (1974) characterized intersection graphs of subtrees of trees as the chordal graphs to provide a possible abstract context for this problem. Meacham (1983) applied these concepts directly to the problems of character compatibility analysis and defined the related concept of partial binary factors. McMorris et al (1994), Argawala and Fernandez-Baca (1994), and Kannan and Warnow (1995) discuss triangulating vertex colored graphs, and give results that seem to recognize whether group-wise potential incompatibilities among QTCs exist.

Currently, enormous quantities of DNA/RNA base pair, and amino acid, sequence

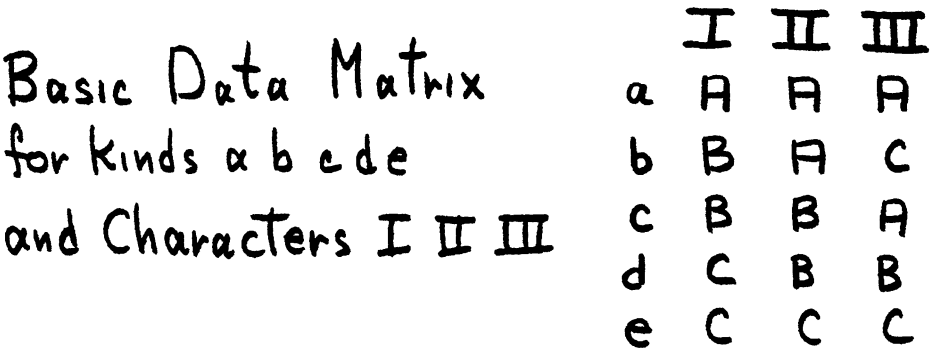


Figure 18a.

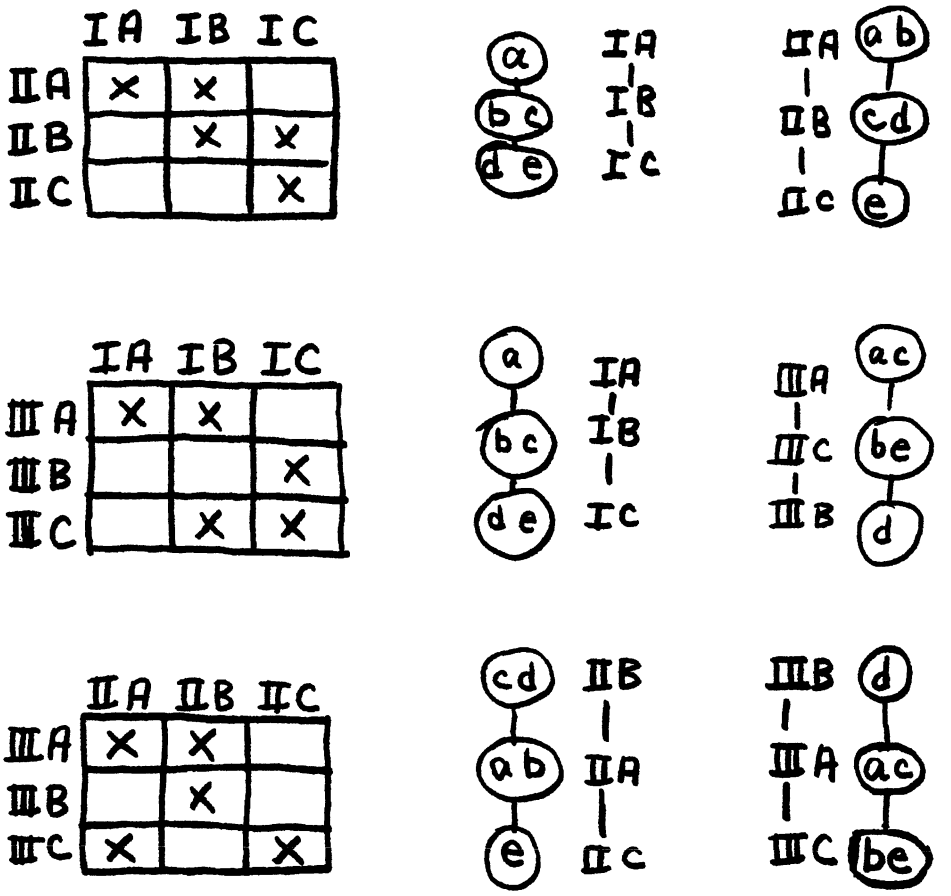


Figure 18b.

data are being produced by molecular biologists. For such data, virtually no basis in an understanding of natural history, development, physiological function, etc. exists for hypothesizing proximity's or direction for CSTs, so these data remain as QTCs. However, a QTC still retains content as a hypothesis about the Ancestor relation: a QTC is *false* if no CST realized with its states is an order homomorphic image of the Ancestor relation. The states of a *true* QTC can be extended to include the unobserved

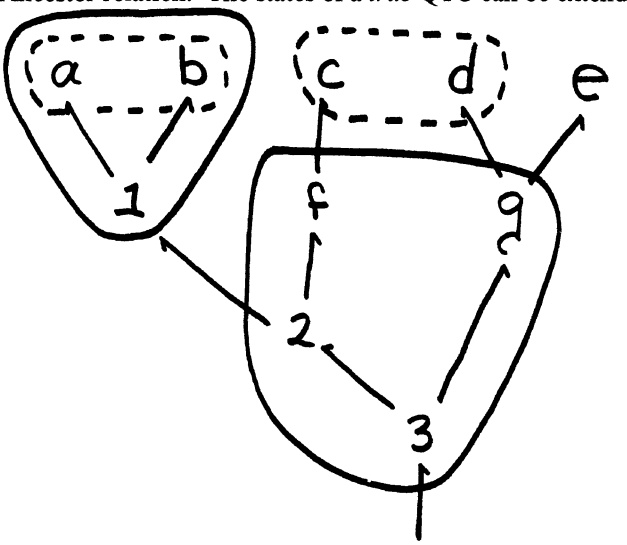


Figure 19.

species in the Ancestor relation so that they are all convex on the Hasse diagram of the Ancestor relation. This means that between any two species in such an extended state there exists an undirected path of covers from one to the other that includes only species in this extended state. This is illustrated in Figure 19., which shows the Hasse diagram for part of an ancestor relation where letters indicate observed species, numbers indicate unobserved species, solid lines enclose convex groups, and dotted

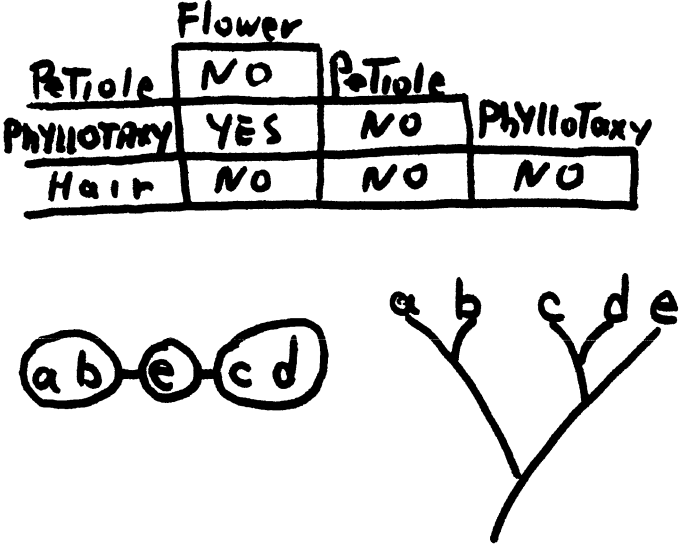


Figure 20.

lines enclose non-convex groups. False QTCs have one or more state that is non-convex in this sense. A better understanding of the nature of the contradictions in groupwise incompatible QTCs would be a significant contribution. A QTC counterpart of the refinement lower semi-lattice of CSTs requires definition and study.

3. Rehypothesizing Characters to Resolve Incompatibility

One approach to rehypothesizing characters is to do nothing. Only a few historically true characters can construct a fairly refined lower bound for the Ancestor relation. There may not be any historically true characters (Baum and Estabrook, 1978), but if there are, they will all belong to the same set of mutually compatible characters, whose sum lies below the Ancestor relation. Thus, the sum of any set of mutually compatible characters is a partial estimate of the Ancestor relation. Typically, in natural data the vast majority of pairs of characters are not compatible. In Figure 6, only one of the six pairs of characters is compatible. The sum of these two compatible characters gives the undirected tree of Figure 20, which also shows a phylogenetic tree consistent with its cep ordering.

There may be several distinct sets of mutually compatible characters among those structured by a comparative biologist for the estimation of the Ancestor relation, the sums of the characters in each producing distinct estimates. How can we choose among them? Biological considerations, such as functional dependencies or parallel selective pressures, might offer non-historical explanations for the observed compatibility of some of the sets, suggesting a choice among the remaining sets. When one of these has been used to estimate an Ancestor relation, this estimate may suggest plausible ways to further divide states into two or more smaller ones convex on the Hasse diagram of this estimate to reflect this parallel selection in rehypothesized compatible characters. These considerations of parallel evolution are best carried out by biological specialists, guided by patterns of compatibility revealed by analysis.

Another reason why characters might be historically incorrect is the observed structures on which the comparisons are based might not be homologous. Structure X in species *a* is homologous with structure Y in species *b* if there is a single structure Z in the most recent ancestor species of *a* and *b* from which both X and Y evolved. Figure 21 shows such homologous structures on the left, but on the right shown non-homologous structures evolved from two different structures in the most recent ancestor species of *a* and *b*. Typically, when closely related species are compared, homologous structures look similar and non-homologous structures look different. Mistakes in recognizing homology occur when non-homologous structures look the

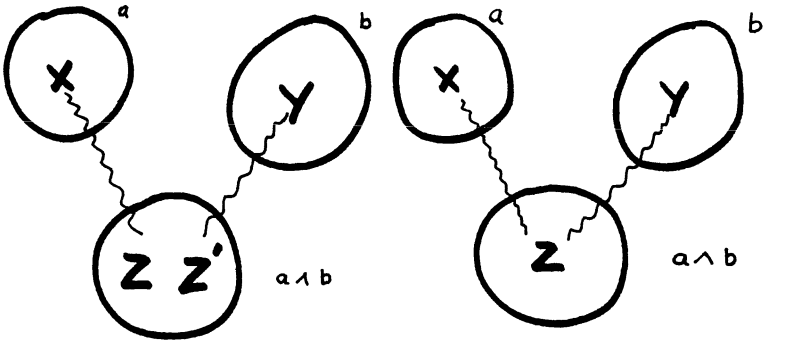


Figure 21.

similar, called convergence, and (less often) when homologous structures look radically different, after much evolutionary change. Figure 22 shows how this might happen in the evolution of a morphological feature: if only species *a*, *b*, *c*, and *d* are not yet extinct and still available for observation, then the distinction short Vs long petals would yield a character historically false (short petal state not convex). The short petals of *a* are not homologous with the short petals of *d*, because in the evolution of the immediate descendant of *a* there was a duplication of petals that was not detected in the comparison of petals later. Perhaps a careful study of the development of petals in the flower bud would suggest this duplication and provide a basis for resolving some of the incompatibilities this character might show with others.

To correct non-convex (false) character states that arise from comparing non-homologous structures it is helpful to understand what evolutionary changes resulted

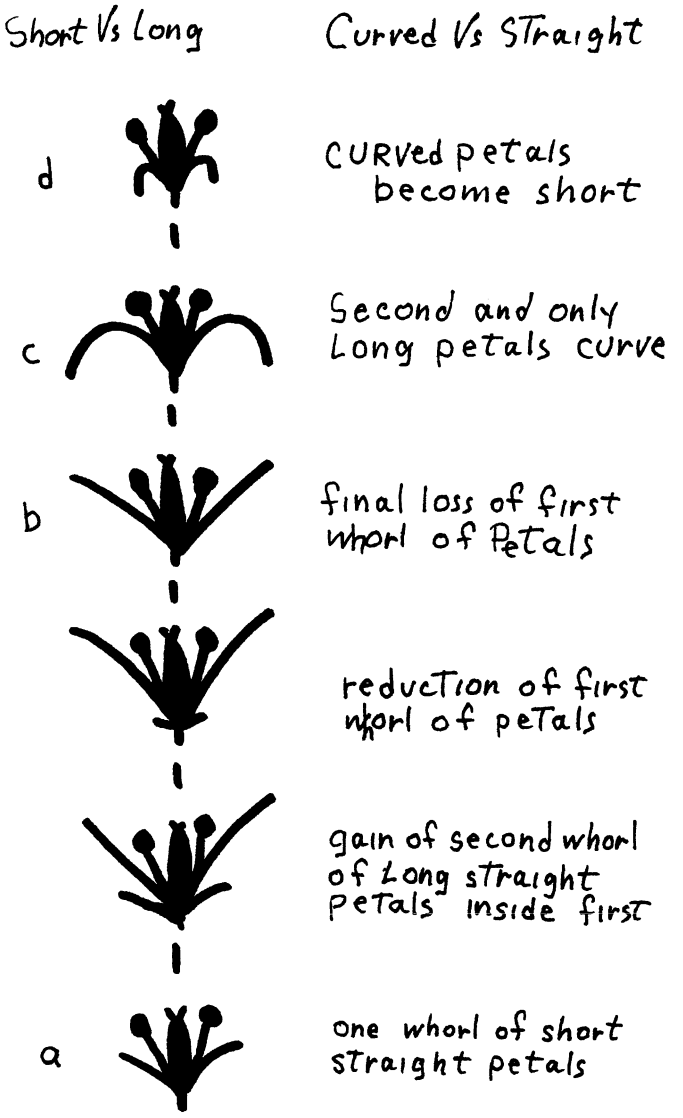


Figure 22.

in the misleading similarity. Sometimes distinct, clearly non-homologous structures in ancestors come to look similar as a result of selection to use them in similar ways. For example, the "leaves" of monocots (like grass) evolved from the petioles of their dicot (like magnolia) ancestors, from which the leaf part has been lost. The ancestral structures are not similar, unlike the case of duplication in which the ancestral structures are initially quite similar prior to evolutionary change.

Duplication and subsequent independent change and loss of parts can also occur at the molecular level, where duplicate segments of DNA or RNA are incorporated into the strand and subsequently replicated in daughter cells. When this happens in reproductive cells, these replicated copies are passed on to progeny and become available for separate evolutionary change futures (Ohno, 1970). Until the two copies have evolved into recognizably distinct entities, homology mistakes are likely to be

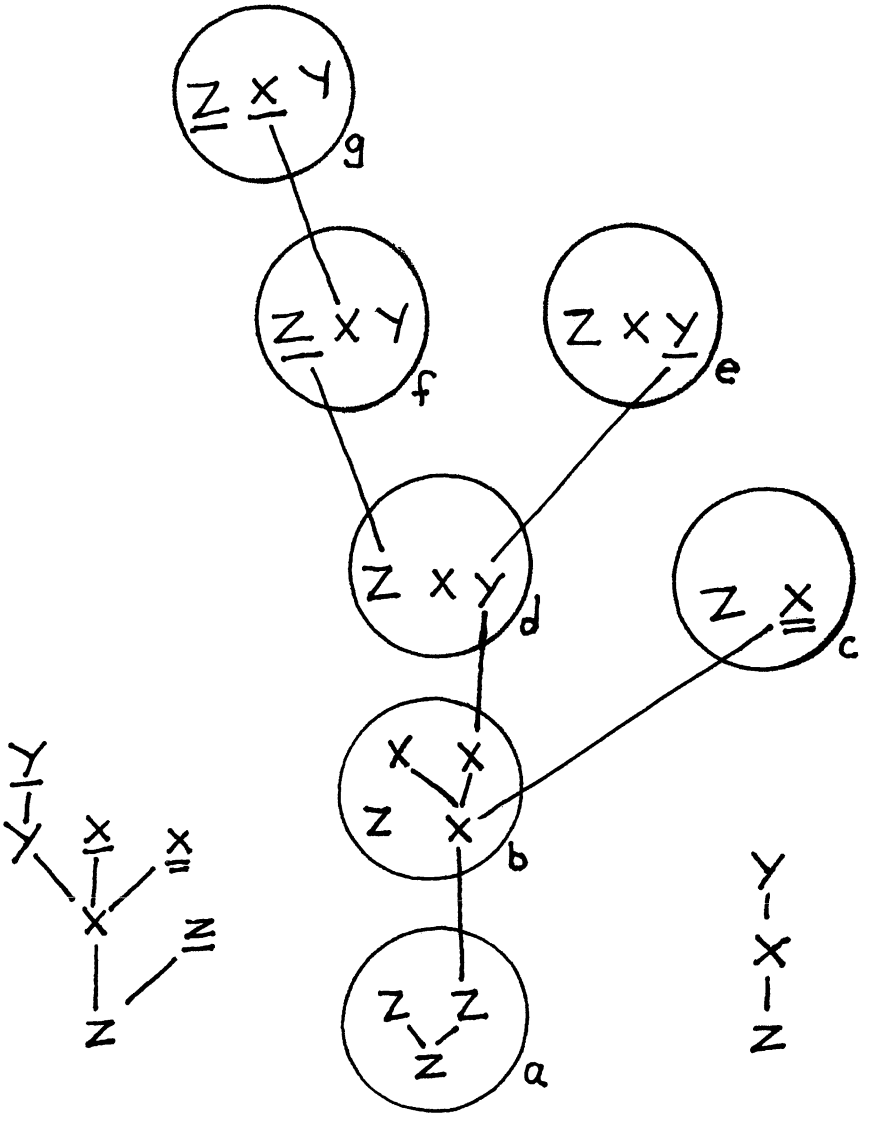


Figure 23.

made: are there two copies? if yes, which copy in one species to compare with which copy in another?; if no has one been lost? or were there never two copies in the ancestry? Figure 23 illustrates this process. Here 7 species are shown by circles labeled *a* through *g*. Inside the circles are shown some genes labeled X, Y, Z; underlined and double underlined indicate the result of evolutionary changes in genes. Within species *a* and *b* are shown gene duplication events. Lines connecting species show the phylogenetic continuity of changing genes in the Ancestor relation of species. The Ancestor relation for genes is in the lower left, and that for gene families lower right. Each gene family gives rise to a CST corresponding to the change events for the genes in that family, shown in Figure 24a. The sum of these three CSTs is the Ancestor relation. Figure 24b shows the case when species *b* and *d* are unobserved ancestors. If no mistakes in homology are made in recognizing which

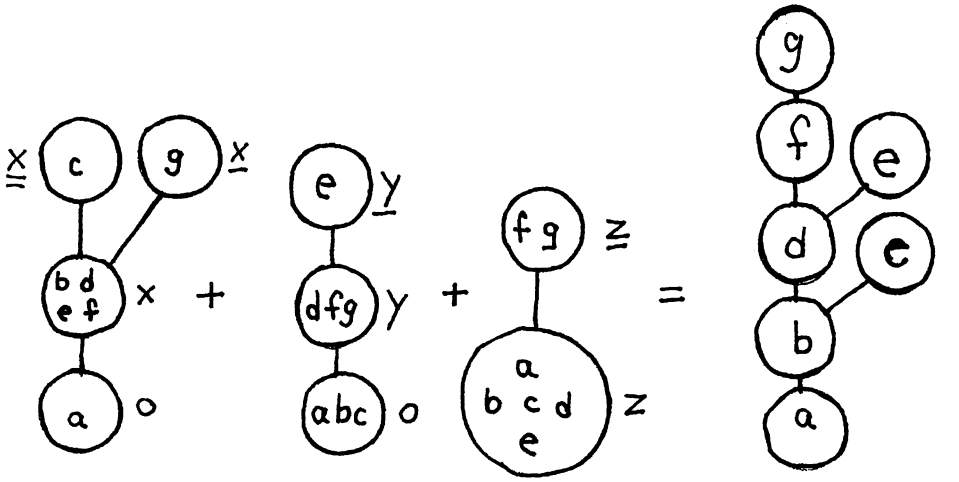


Figure 24a.

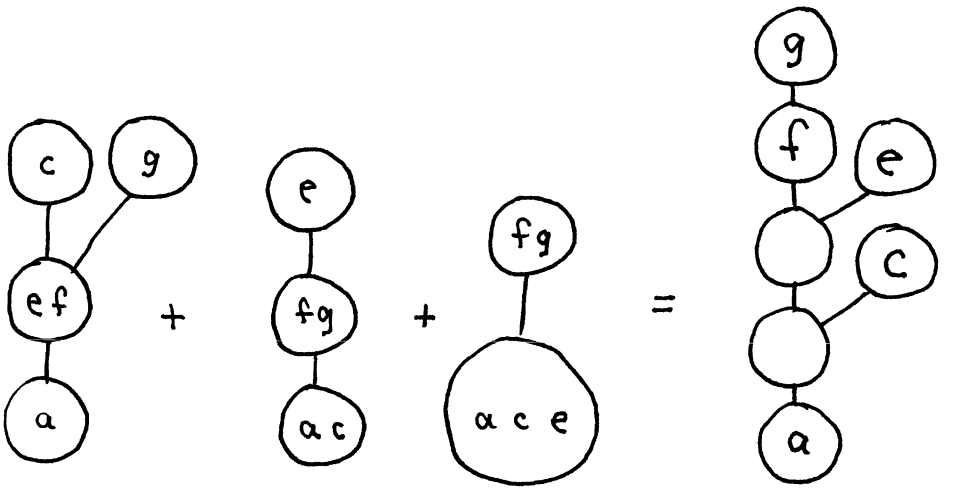


Figure 24b.

genes belong to which families, then their corresponding QTCs are sufficient to determine the branching of the species tree. But, for example, mistaking X double underline as a member of the Y family would suggest that c and a were both in the "not yet evolved" state for character X, which would make this a nonconvex state. This mistake would be especially dangerous because it would result in compatible characters that would determine an incorrect ancestor relation.

The problem of aligning sequences of bases comprising parts of DNA or RNA molecules is a problem in estimating homologous correspondence. Once sequences representing the species under study have been aligned, this alignment determines a QTC for each position by placing together in the same state all species with the same base in that position. These QTCs could be false if the alignment contains mistakes in homology. For transcribing regions, aligning the corresponding amino acid sequences helps reduce errors in homology, as does a consideration of the secondary and tertiary structure of the resulting protein, but in most cases biological considerations cannot dissipate all uncertainty. Needleman and Wunch (1970) and Waterman and Smith (1981) were among the first to propose algorithms to automatically align sequences to produce hypotheses of homologous correspondence. These algorithms try to make sequences look as similar as possible by introducing gaps to slide amino acids into new positions, and by manipulating sequences in other ways as well, so to align positions with the same amino acids. A variety of rules to measure the success of minimizing gaps and manipulations while maximizing the similarity among sequences has produced a variety of criteria. Many have been implemented in clever algorithms to analyze large data sets. McClure et al. (1994) and Day and McMorris (1994) critically compared some of them.

These algorithms often do an excellent job of finding and aligning the invariant positions to produce credible global alignments that concentrate uncertainty in a few highly variable local areas. Although the invariant positions are important to enable the estimation of global alignments by maximizing similarity, the variable positions, where estimates of homology are least certain, are the ones potentially useful for estimating the Ancestor relation. Maximizing compatibility, instead of similarity, to align these remaining variable positions, may hypothesize homologous correspondence in a more biologically realistic way. How to do this poses a problem for discrete mathematics to solve.

Meacham (1981) suggested that if any permutation of the species among the character states is equally likely then a character could be considered random. Using random CSTs with the same state tree orders and same numbers of species in each state, he calculated the probability that a given set of characters be mutually compatible at random. A mutually compatible set of characters likely to be compatible at random may not indicate history. Recently, Meacham has enriched his combinatoric procedures with powerful simulating techniques to enable the analysis of data sets with hundreds of QTCs such as are now commonly generated by automatic protein or DNA/RNA sequencing machines, and to calculate the distributions of a random variable associated with each character, whose value is the number of remaining characters with which it is potentially compatible. Characters with observed potential compatibility counts typical of random values may be considered unlikely to reflect history. If there is no basis for correcting them, they can be confidently ignored. This analysis has recently been applied by Meacham (1994) to lower Angiosperms, and by Camacho et al. (1997) to Crustacea.

4. Automatic Character Restructuring to Resolve Incompatibility

When patterns of compatibility have been considered, additional understanding with which to rehypothesize characters has been exhausted, apparently random characters eliminated, and no reasonable estimate of the Ancestor relation has been achieved, biologists can give up and turn the problem of estimating the Ancestor relation over to a minimizing paradigm, of which there are many. The idea is to reconcile all the remaining incompatibilities by doing some ad hoc thing as infrequently as possible (parsimony).

Algorithms to use parallel and/or reversed evolution to automatically rehypothesize characters by breaking states into as few as necessary smaller states, under a variety of constraints, have been distributed in the popular computer package PAUP (Swofford, 1993), but many other programs to do this exist, and much has been written about these algorithms and their applications to data. Sometimes these algorithms generate a large number (thousands) of estimates of the Ancestor relation, which need to be consolidated or themselves reconciled before they can be grasped. This poses a problem for discrete mathematics. Margush and McMorris (1981) presented some of the earliest rigorous results. Many people question the historical credibility of automatic parsimony algorithms, finding it difficult to believe that evolution would favor one change over another in anticipation of reducing the number of parallel or reversed changes millions of years later (Meacham and Estabrook 1985), but their use may be justified in cases where we have no other basis for judgment. Such algorithms are widely used at present.

Incompatibilities may be resolved, and perhaps truth revealed, also by hypothesizing different homologous correspondence among the structures being compared. These revisions in characters are best based on explicit biological considerations, as discussed above. When biological knowledge is exhausted, the use of algorithms to resolve incompatibilities by automatically hypothesizing different homologies may be justified. Goodman et al. (1979) described an algorithm to automatically rehypothesize homologous correspondence among genes that attempted to minimize events of gene duplication and subsequent loss. Page (this symposium) and Mirkin et al. (1995) more recently describe algorithms of this kind.

The view of homologous structures, such as petals (Fig 22) or genes in organisms (Fig 23), can be generalized to include homologous parasites in/on organisms, and, in the view of some authors such as Nelson (1983), homologous species in evolving floras or faunas. When these generalizations are carefully defined so that real biological meaning (not just algorithmic mechanics) can be applied to the concepts, resolution of conflict in hypothesized characters by estimating homologous correspondence accordingly can be a powerful tool for the study of co-evolution. Page (1994, 1996) describes studies applying these concepts and techniques. Identifying and localizing character incompatibilities in this context, and understanding the discrete structures associated with them would enable biologists to resolve conflicts with biological information and considerations to the extent possible, before turning to an automatic algorithm.

5. Variation in Paradigm

The details of the paradigm defining characters and their historical truth, and consequently defining logical compatibility and associated discrete structures, have

been chosen to reflect a particular form of data and body of biological concepts. In some cases, variations in these details may be exigent and may give rise to different structures with interesting properties to discover.

Additional constraints arise from knowledge of fossils. When data reliably indicate the past time span during which the species under study lived, any hypothesized ancestor relation that contains a descendant that evolved before one of its ancestors, or after its immediate ancestor had gone extinct, is false. In some studies, some geologists consider the stratigraphic data of paleontologists to be reliable for this purpose (Fisher, 1994; Huelsenbeck, 1994; Wagner, 1995). Recall that two characters are compatible if they are each structure preserving (homomorphic) images of the same possible ancestor relation. Because known time spans for species eliminate many possible ancestor relations, they may also convert formerly compatible characters to incompatible ones. Individual characters may be incompatible with the time span data. These concepts are illustrated in Fig 25, where dark vertical lines represent the time spans of the species whose labels are below them. Character I is false because its left state is not convex; Character II is true; and as a cladistic character, III violates direction in the transition to its upper right state.

If fossil records are especially thoroughly sampled, workers may consider it unlikely that unobserved ancestors actually existed and so disqualify ancestor relations that contain them (Alroy, 1995; Huelsenbeck, 1991). As argued earlier, disqualified ancestor relations reduce compatibilities, and create new, uninvestigated discrete mathematical structures. Indeed, for explaining away incompatibilities, creating unobserved ancestors may seem to be as ad hoc as the otherwise unjustified subdivisions of states that form the basis for the popular parsimony algorithms discussed earlier. Fisher (1994), among others, has suggested adding penalties for

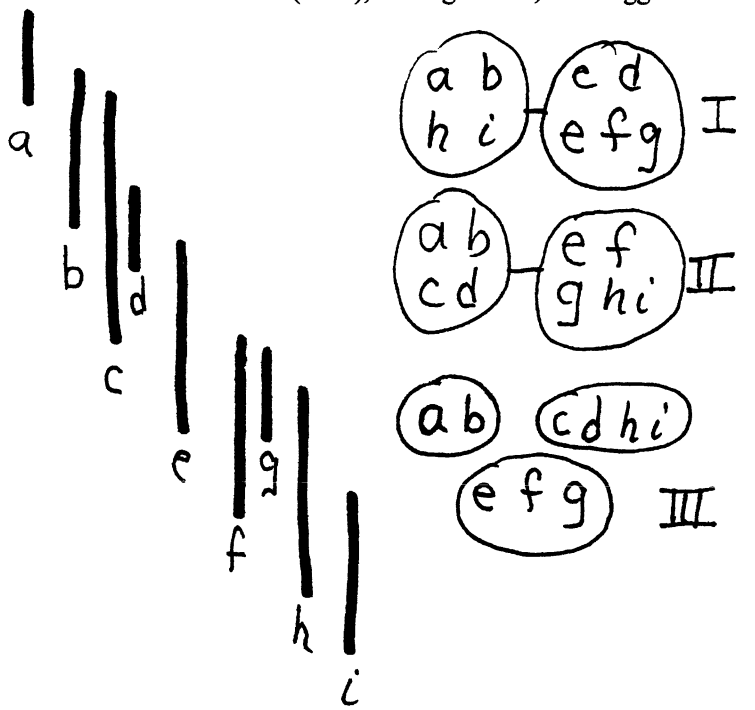


Figure 25.

using species that did not exist when they were needed (or ever) to penalties for subdividing states to make a minimizing criterion for parsimony algorithms that would take this into consideration.

However, the elimination of unobserved ancestors from Ancestor relations is much more interesting than its use as a parsimony criterion, because this makes it possible for one-state characters to contradict each other. A one-state character is a subset of the species under study determined by the possession of a feature; this subset, enlarged by the inclusion of any unobserved ancestors also possessing this feature, is hypothesized to be convex on the Ancestor relation. The character is one-state because no hypothesis is advanced about the rest of the species. An example is the evolution of restriction enzyme recognition sites. These are short, complex DNA sequences recognized by enzymes, in which any change results in nonrecognition. Molecular biological techniques reveal whether a species has a given site. The site is likely to have evolved only once but have been easily lost many times. Therefore, the subset of species with the site (together with their ancestors with the site) may be convex, but species lacking the site (together with their ancestors lacking the site) are not so likely to be. (Templeton et al, 1992) provide an example of this kind of data.

When unobserved ancestors can be assigned to any state required, then for any collection of one-state characters there is always an Ancestor relation on which they are all convex. Because they are not contractible in this sense, they assert nothing and so are useless for estimating the Ancestor relation. Minimizing the number of times one-state properties are lost provides an ad hoc parsimony criterion, called Dollo parsimony, for estimating the Ancestor relation (Felsenstein 1979, 1983) with one on which all the one-states are convex. When unobserved ancestors are not permitted in the Ancestor relation, then one-state characters can contradict each other, which gives rise to a potentially interesting and largely unstudied body of discrete structures.

One view of incompatible characters looks at unobserved ancestors as the site of the conflict, where some such ancestors would have to belong to two (or more) states of the same character, were all the states to have convex extensions on any estimate of the Ancestor relation. One way to resolve these incompatibilities is to allow some unobserved ancestors to belong (in some sense) to both states at the same time, i.e., be polymorphic. Such polymorphisms can always resolve incompatibilities, often in a wide variety of ways. Thus, without some biological justification for some and not others, polymorphic resolutions are not biologically compelling. However, if no further biological evidence is available, estimating the Ancestor relation by choosing one that minimizes the number of polymorphisms provides yet another ad hoc parsimony criterion, discussed by Felsenstein (1979) and Benham et al. (1995). Bonet et al. (in review) present algorithms to determine the existence of an ancestor relation that resolves all incompatibilities for a given collection of characters by including unobserved ancestors that belong to no more than a given number of states of the same character, and when such exists to produce an example.

Finally, we can allow the Ancestor relation to include more than one immediate ancestor for some species. Such might be the case if one species evolved as the result of hybridization between two other species. The Ancestor relation shown in Figure 26 has two immediate ancestors for species *e*. Also shown are two QTCs that are incompatible for a tree ancestor relation, but whose states are all convex on the Ancestor relation shown. In fact, any incompatibility can be resolved by hypothesizing hybridizations, which would seem uninteresting therefor, unless constrained somehow.

Biological constraints are compelling, when available. Corti et al. (1986) described mice whose genomic configurations made it possible for only a small fraction to hybridize. They hypothesized hybridization to resolve incompatibilities only when it was genomically possible. It is possible that hybridization occurred in the distant past during the evolution of an unobserved ancestor. For example, within the rose family, apples and their relatives may have evolved from a hybrid between a plum-like ancestor and a spiraea-like ancestor, millions of years ago (Campbell et al. 1995). If allowing one hybridization in the Ancestor relation resolves a surprisingly large number of incompatibilities in characters of the rose family, then this would support this hypothesis. Ancestor relations with one hybridization can be enumerated and displayed as functional directed graphs (Harary et al. 1965), which may provide a first step toward a better understanding of one hybrid incompatibility.

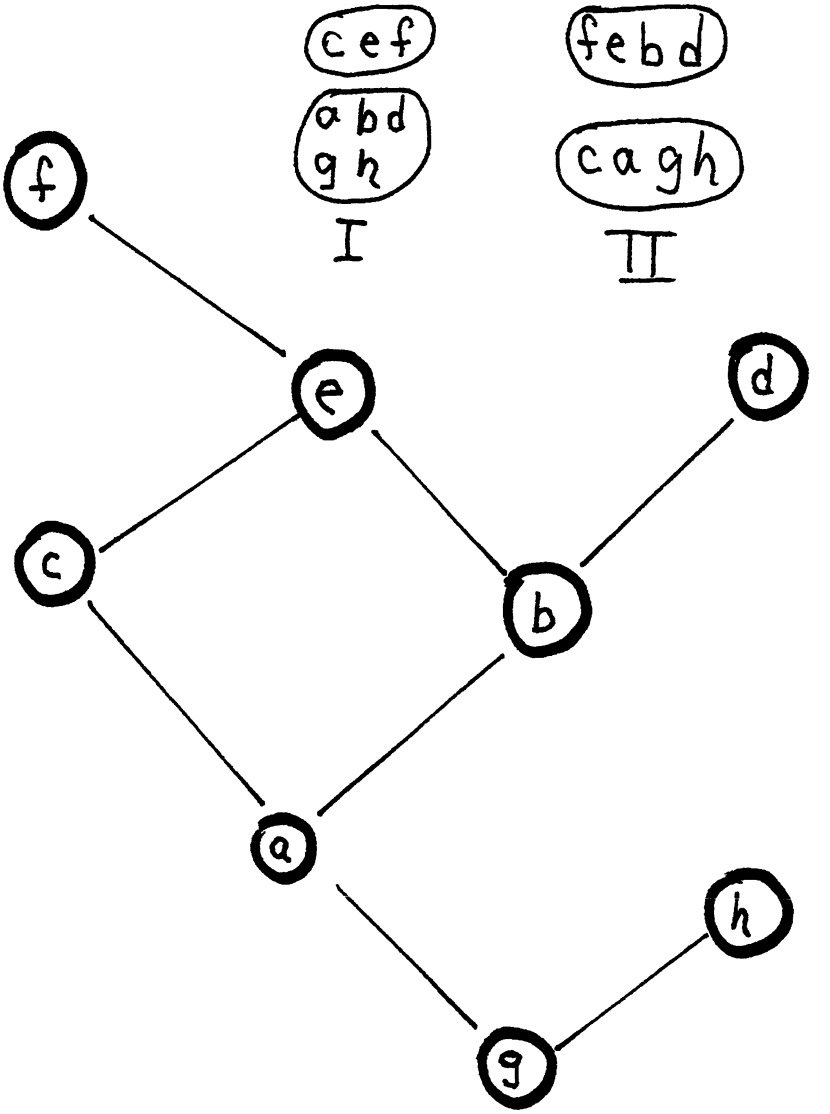


Figure 26.

Alroy (1995) proposes to admit hybrids in the Ancestor relation but not unobserved ancestors. He then proceeded to construct an estimate from one-state characters by including a fixed number of edges in the diagram of the ancestor relation in such a way as to minimize the number of convex pieces of character states. He suggests that the purpose of such a diagram may not be so much to estimate the true Ancestor relation as to reveal the patterns of compatibility and incompatibility in the data.

In this spirit, it may be interesting to resolve all incompatibilities through the introduction of sufficient cycles in the graph of the ancestor relation, as a means of revealing the patterns of incompatibility for the edification of scientists. In the case of two-state characters, Bandelt (1994) has proposed median networks for this purpose, and Bandelt et al. (1996) have applied the concepts to a study of mitochondrial diversity within an Amerindian tribe (Ward et al. 1991).

It is interesting to note that revealing patterns of incompatibility for the edification of scientists, who are thereby aided to use whatever they know about their science to explain or resolve them, was a primary goal of original research during the 70's on the discrete structures of compatibility analysis. The computer programs written to implement these results (Fiala, 1984) explicitly exhibit incompatibility cycles for multi-state character state trees. I believe that the primary potential for future contributions of discrete mathematics to the estimation of the Ancestor relation by biological scientists centers on the elucidation of the structure of incompatibilities in the data, so that these can be better resolved or explained by scientific means, and does not center so much on algorithms to automatically resolve incompatibilities when science fails to do so.

6. References

1. Alroy, J. Continuous track analysis: a new phylogenetic and biogeographic method. Systrematic Zoology **44** (1995), 152-178.
2. Argawala, R., D. Fernandez-Baca. Fast and simple algorithms for the perfect phylogeny problem and triangulating colored graphs. DIMACS Technical Report (1994), 94-51.
3. Argawala, R., D. Fernandez-Baca and G. Slutzki. Fast algorithms for inferring evolutionary trees. J. Computational Biology **2** (1995), 397-407.
4. Bandelt, H-J. Phylogenetic networks. Verh. naturwiss. Ver. Hamburg (NF) **34** (1994), 51-71.
5. Bandelt, H-J., P. Forster, B. C. Sykes, and M. B. Richards. Mitochondrial portraits of human population using median networks. Genetics **141** (1995), 743-753.
6. Baum, B. R. and G. F. Estabrook. Application of compatibility analysis in numerical cladistics at the infraspecific level, Canadian J. Botany **56** (1978), 1130-1135.
7. Benham, C., S. Kannan, M Paterson and T. J. Warnow. Hen's teeth, and whale's feet: generalized characters and their compatibility. J. Computational Biology **2** (1995), 515-525.
8. Benson, L, Plant Classification. Heath & Co, Boston, 1957, pp. xvi+688.
9. Bessey, C. E. 1915. The Phylogenetic Taxonomy of Flowering Plants. Annals of the Missouri Botanical Garden **2** (1915), 001-118.
10. Bonet, M., C. Phillips, T. J. Warnow, and S. Yooseph. Constructing evolutionary trees in the presence of polymorphic characters. Proceedings of the 1996 IEEE symposium on the theory of computing (1996), 220-229.
11. Boulter, D., D. Peacock, A. Guise, J. T. Gleaves, and G. Estabrook. Relationships between the partial amino acid sequences of plastocyanin from members of ten families of flowering plants. Phytochemistry **18** (1979), 603-608.
12. Camacho, A. I., E. Bello and G. F. Estabrook. A statistical approach to the evaluation of characters to estimate evolutionary history among the species of the aquatic subterranean genus, Iberobathynella (Crustacea, Snycardia). Biological J. Linnean Society **60** (1997), 221-241.
13. Camin, J. and R. Sokal. A method for deducing branching sequences in phylogeny, Evolution **19** (1996), 311-326.

14. Corti, M., E. Capanna, and G. F. Estabrook. Microevolutionary sequences in house mouse chromosomal speciation. Systematic Zoology **38** (1986), 163-173.
15. Cronquist, A. The Evolution and classification of Flowering Plants. Houghton Mifflin, Boston, 1969, pp. xi+396.
16. Day, W. H. E. and F. R. McMorris. Alignment, comparison, and consensus of molecular sequences. in New Approaches in Classification and Data Analysis. E. Diday et al. eds. Sprenger Verlag Berlin, 1994, pp. 327-346.
17. Estabrook, G. F. A general solution in partial orders for the Camin-Sokal model in phylogeny. J. Theoretical Biology **21** (1968), 421-438.
18. Estabrook, G. F. Cladistic methodology. Annual Reviews of Ecology and Systematics **3** (1972), 427-456
19. Estabrook, G. F., C. S. Johnson and F. R. McMorris. An idealized concept of the true cladistic character. Mathematical BioSciences **23** (1975), 263-272.
20. Estabrook, G. F., C. S. Johnson and F. R. McMorris. An algebraic analysis of cladistic characters. Discrete Mathematics, **16** (1976a), 141-147.
21. Estabrook, G. F., C. S. Johnson and F. R. McMorris. A mathematical foundation for the analysis of cladistic character compatibility. Mathematical BioSciences **29** (1976b), 181-187.
22. Estabrook, G. F. and F. R. McMorris. When are two qualitative taxonomic characters compatible? J. Mathematical Biology **4** (1977), 195-200.
23. Estabrook, G. F., J. G. Strauch, Jr. and K. L. Fiala. An application of compatibility analysis to the Blackiths' data on Orthopteroid insects. Systematic Zoology **26** (1977), 269-276.
24. Estabrook, G. F. Some concepts for the estimation of evolutionary relationships in systematic botany. Systematic Botany **3** (1978), 146-158.
25. Estabrook, G. F. and W. R. Anderson. An estimate of phylogenetic relationships within the genus Crusea (Rubiaceae) using character compatibility analysis. Systematic Botany **3** (1978) 179-196.
26. Estabrook, G. F. and C. A. Meacham. How to determine the compatibility of undirected character state trees. Mathematics BioSciences **46** (1979) 251-256.
27. Estabrook, G. F. and F. R. McMorris. When is one estimate of evolutionary relationships a refinement of another? J. Mathematics Biology **10** (1980) 367-373.
28. Estabrook, G. F. Phylogenetic trees and character state trees. in Cladistics (T. Duncan and T. F. Stuessy eds.), Columbia University Press, New York, NY, 1984, pp. 135-151.
29. Felsenstein, J. Alternative methods of phylogenetic inference and their interrelationship. Systematic Zoology **28** (1979), 49-62.
30. Felsenstein, J. Parsimony in systematics: biological and statistical issues. Annual Review Ecology Systematics **14** (1983), 313-333.
31. Fiala, K. L. CLINCH: Cladistic Inference by Compatibility of Hypothesized Character State Trees. FORTRAN program and User's Document. Museum of Zoology, University of Michigan, Ann Arbor. 1984.
32. Fisher, D. C. 1994. Stratocladistics: morphological and temporal patterns and their relation to phylogenetic process. pp. 133-171 in Interpreting the Hierarchy of Nature. L. Grande and O. Rieppel, eds. .Academic Press. San Diego. CA USA.
33. Fitch, W. M. Towards finding the tree of maximum parsimony. in Estabrook, G. F. ed. Proceedings of the Eighth International Conference on Numerical Taxonomy, Freeman. San Francisco, 1975, pp. 189-230.
34. Gavril, F. The intersection graphs of subtrees in trees are exactly the chordal graphs. J. Combinatorial Theory B **16** (1974), 47-56.
35. Goodman, M., J. Czelusniak, G. W. Moore, A. E. Romerro-Herrera, and G. Matsuda. Fitting the gene lineage into its species lineage, A parsimony strategy illustrated by cladograms constructed from globin sequences. Systematic Zoology **28** (1979), 132-162.
36. von Haeseler, A. and G. A. Churchill. Network models for sequence evolution. J. of Molecular Evolution **37** (1993), 77-85.
37. Hall, H. M. and F. E. Clements. The Phylogenetic Method in Taxonomy: the North American Species of Artemesia, Chrysothamnus and Atriplex. Carnegie Institution, Washington, DC, 1923, Fig 28 pp. 223.
38. Harary, F., R. Z. Norman and D. Cartwright. Structural Models: An Introduction to the Theory of Directed Graphs. Wiley, New York., 1965.
39. Hennig, W. Phylogenetic Systematics. (Translated from German manuscript by D. D. Davis and R. Zangerl), University of Illinois Press , 1966, pp. 263.
40. Huelsenbeck, J. P. Comparing the stratigraphic record to estimates of phylogeny. Paleobiology **1** (1994), 470-484
41. Huelsenbeck, J. P. When are fossils better than extant taxa in phylogenetic analysis? Systematic Zoology **40** (1991), 458-469.
42. Kannan, S. and Warnow, T. Inferring evolutionary history from DNA sequences, SIAM J. Comput. **13** (1995), 338-335.
43. LeQuesne, W. J. A method of selection of characters in numerical taxonomy. Systematic Zoology **18** (1969), 201-205.

44. Meacham, C. A. A probability measure for character compatibility. Mathematical BioSciences **57** (1981), 1-18.
45. Meacham, C.A. Theoretical and computational considerations of the compatibility of qualitative taxonomic characters. Numerical Taxonomy. Felsenstein, J. (ed) NATO Adv Study Inst. Ser. Vol. G1. Springer-Verlag Berlin, 1983, pp.304-314.
46. Meacham, C.A. The role of hypothesized direction of characters in the estimation of evolutionary history. Taxon **33** (1984a), 26-38.
47. Meacham, C. A. Evaluating characters by character compatibility analysis. in Cladistics T. Duncan and T. F. Stuessy (eds.). Columbia University Press. New York, NY, 1984b, pp. 135-151.
48. Meacham, C. A. and G. F. Estabrook. Compatibility methods in Systematics. Annual Review of Ecology and Evolution **16** (1985), 431-446.
49. Meacham, C. A. Hasse diagrams of the "is a refinement" semi- lattice, Classnotes, University of California, Berkeley, 1989.
50. Meacham, C. A. Phylogenetic relationships at the basal radiation of Angiosperms: further study by probability of compatibility. Systematic Botany **19** (1994) 506-522.
51. McClure, M. A., T. K. Vasi and W. M. Fitch. Comparative analysis of multiple protein-sequence alignment methods. Molecular Biological Evolution **11** (1994), 571-592.
52. McMorris, F.R. On the compatibility of binary qualitative taxonomic characters. Bull. Mathematics. Biol. **39** (1977), 133-138.
53. McMorris, F. R. and T. Zaslavsky. The number of cladistic characters. Mathematical BioSciences **54** (1981), 3-10.
54. McMorris, F. R., T. Warnow and T. Wimer. Triangulating vertex colored graphs, SIAM J. Discrete Mathemetics **7** (1994), 296-306.
55. Mirkin, B., I. Muchnik and T. F. Smith. A biologically consistent model for comparing molecular phylogenies. J. Computational Biology **2** (1995), 493-507.
56. Needleman, S. B. and C. D. Wunsch. A general method applicable to the search for similarities in the amino acids sequences for two proteins. J. Molecular Biology **48** (1970) 443-453.
57. Nelson, G. 1983. Cladistics and biogeography. in T. Duncan and T. F. Stuessy (eds). Cladistics, Columbia University Press. New York, NY pp. 273-293.
58. Ohno, S. 1970. Evolution by Gene Duplication. Sprenger-Verlag Bonn.
59. Page, R. D. M. Maps between trees and cladistic analyses of historical associations of genes, organisms, and areas. Systematic Biology **43** (1994) 58-77.
60. Page, R. D. M. Temporal congruence revisited: comparison of mitochondrial DNA sequence divergence in cospeciating pocket gophers and their chewing lice. Systematic Biology **45** (1996) 151-167.
61. Skelton, P. W. Adaptive radiation: definition and diagnostic test. in Lees R.D. and D. Edwards (eds.) Evolutionary Patterns and Processes. Linnean Society Symposium Series, **14** (1993), 46-58.
62. Smith, T. F. and M. S. Waterman. Identification of common molecular subsequences. J. Molecular Biology **17** (1981) 195-197.
63. Sneath, P. H. A., M. J. Sackin, and R. P. Ambler. Detecting evolutionary incompatibilities from protein sequences. Systematic Zoololgy **24** (1975), 311-332.
64. Strauch, J. G. Jr. The use on homoplastic characters in compatibility analysis. Systematic Zoolology **33** (1984), 167-177.
65. Swofford, D. L. PAUP: Phylogenetic Analysis Uning Parsimony. version 3.1 Illinois Natural History Survey, Champaign, IL, 1991.
66. Takhtajan, A. Flowering Plants. (Translated from Russian by C. Jeffrey) Smithsonian Institution Press, Washington DC, 1969, pp. x+310.
67. Templeton, A. R., K. A. Crandall, and C. F. Sing. A cladistic analysis of haplotypes inferred from restriction endonuclease mapping and DNA sequence data. III. Cladogram estimation. Genetics **132** (1992), 619-633.
68. Waterman, M. S. Efficient sequence alignment algorithms, J. Theoretical Biology **1108** (1984), 333-337.
69. Wagner, P. J. Stratigraphic tests of cladistic hypotheses. Paleobiology **2** (1995), 153-178.
70. Ward, R. H., B. L. Frazer, K. Dew-Jager, and S. Paabo. Extensive mitochondrial diversity within a single Amerindian tribe. Proc. National Acad. Sciences USA **88** (1991), 8720-8724.
71. Wilson, E. O. A consistency test for phylogenies based on contemporary species. Systematic Zoology **14** (1965), 214-220.

DEPARTMENT OF BIOLOGY, THE UNIVERSITY OF MICHIGAN, ANN ARBOR, MI 48109-1048
E-mail address: Estabrook@umich.edu
Web-page: http://www-personal.umich.edu/~gfred/

https://doi.org/10.1090/dimacs/037/02

DIMACS Series in Discrete Mathematics
and Theoretical Computer Science
Volume **37**, 1997

Krohn-Rhodes Theory, Hierarchies & Evolution

Chrystopher L. Nehaniv and John L. Rhodes

ABSTRACT. We give a natural axiomatization for the notion of hierarchical complexity measures for biological systems modelled by finite-state automata. The algebraic theory of automata is applied to show the existence of a unique maximal complexity measure satisfying these axioms, and relates hierarchical complexity to global semigroup theory. We then study the rate at which hierarchical complexity can evolve in biological systems assuming evolution is "as slow as possible" from the perspective of computational power of organisms. Explicit bounds on the evolution of complexity are derived showing that, while the evolutionary changes in hierarchical complexity are bounded, in some circumstances complexity may more than double in certain 'genius jumps' of evolution. In fact, examples show that our bounds are sharp. We sketch the structure where such complexity jumps are known to occur and note some similarities to previously identified mechanisms in biological evolutionary transitions.

1. Introduction

1.1. Complexity Increase. Biologists generally agree that evolution need not entail "progress", yet at the same time it is clear that the history of life shows there has been *complexity increase* in major evolutionary transitions [**Bon88, MSS95**]. More complex organisms do not supplant the less complex; indeed, despite anthropocentric views, we are living in the age of bacteria, the most successful type of organism, if one measures success by sheer numbers. Nevertheless, for increases in size or complexity there seems to always be "room at the top" in the evolutionary sense, due, in part, to the exploitation of new environmental niches by the innovators [**Bon88**].

This raises the question of how to measure complexity increase, and the logically prior question of rigorously measuring complexity at all. We show that once a complexity measure has been reasonably defined, one can ask (and answer) questions concerning the rate and smoothness of evolutionary complexity increase.

The Cartesian viewpoint leading to the modelling of organisms by finite automata first blossomed in the 1960s with the pioneering work of several independent researchers: Krohn, Langer, and Rhodes [**KLR67, Rho71a**], Lindenmeyer [**Lin68**], and Kauffman [**Kau72**]. The modern mechanistic viewpoint of cellular machinery, environment and genome interacting via feedback control, and the extension of this viewpoint to growth and development, fit within an automata-theoretic viewpoint. Of course research into these areas proceeds now at a furious

1991 *Mathematics Subject Classification.* Primary 92D15, 68Q70, 20M35, 92B05, 20M20; Secondary 03D15, 03D55, 68Q15.

pace, as do efforts to exploit this knowledge for medical, scientific, agricultural and economic gain.

We axiomatize the notion of complexity measures for biological systems. Assuming that living systems can be adequately modelled using finite automata, we find the (unique) integer-valued complexity measure satisfying the axioms. We show that there are mathematically demonstrable bounds on complexity change, and calculate them explicitly. It is perhaps surprising that, even assuming slow-as-possible change in the computational power of organisms, while complexity may fall or rise very slowly, in certain circumstances it may more than double! (In one generation, complexity can go from n to $2n + 1$.) Or conversely, it may fall to slightly less than half its value for an immediate ancestor. These results do not depend on the construction of any *particular* models, but only on the assumption that modelling of organisms as finite automata is possible in principle.

Examples of such 'genius-jumps' of evolution are constructed using techniques of Krohn-Rhodes algebraic automata and semigroup theory establishing the sharpness of the bounds in our forthcoming paper [**NR**]. Work on the nature of evolutionary transitions [**Bus87, Ohn70, DeR96, MSS95**] supports the idea that hierarchical structuring and interaction of previously existing non-interacting or weakly interacting components of biological systems, or the construction of a higher level structure from modified multiple instances of biological subsystems, are common mechanisms of evolutionary jumps in complexity. Our mathematical examples also exhibit these properties. This suggests that innovative re-arrangement of existing components into hierarchical structures may be a generic mechanism of complexity increase in the evolution of all possible biological systems. Perhaps the mathematical principles studied in this paper constrain the nature of evolutionary complexity increase to occur only or primarily in this manner.

1.2. Two Kinds of Hierarchy. The notion of hierarchy emerges here in at least two ways: Firstly, the axiomatization of the idea of the *number of levels needed to describe a biological system* will yield a measure of the *minimum* amount of hierarchical structuring needed in understanding the system — note well that this will *not* capture the number of levels that actually occur within the structure of a particular organism or biological system, nor will knowing this hierarchical complexity answer the very difficult questions of how to identity the levels of structuring, or units of structure, nor even whether they be anatomical, physiological, developmental, ecological or of other other, possibly mixed, natures. Nevertheless, efforts to calculate the hierarchical complexity of a biological system will naturally lead one to ask and refine such questions, but the fact of hierarchical complexity's well-definedness does not depend on the answers to such questions. The hierarchical complexity value is an *intrinsic* property of any finite-state model of the organism.[1]

Secondly, since finite-state models of biological systems each have an intrinsic integer-valued hierarchical complexity, it is natural to consider those whose complexity takes a particular value n as a class. This corresponds to the famous Krohn-Rhodes hierarchy of finite-state automata (or, equivalently, of finite semigroups).

[1] Note that this further implies that the hierarchical complexity value is an intrinsic property of the organism itself, if we define the hierarchical complexity of the organism as the minimum hierarchical complexity over all 'adequate' automata models for the organism. Of course, 'adequate' would have to be formally defined.

In particular, the class of all biological systems that can be modelled by finite-state machines of hierarchical complexity not exceeding n enjoys several algebraic closure properties: colonies of non-interacting members (aggregations of individuals) from the class are also in the class, and those organisms whose representing state-automata can be emulated by organisms or non-interacting aggregations of organisms in a class are also in the class. If interaction occurs between units of the class, then the hierarchical complexity may or may not increase. Thus the second aspect of complexity is comparative, placing a system in relation to others, in a hierarchy of increasingly complex classes of systems.

These properties for a rigorous measure of hierarchical complexity are captured in the complexity axioms below. They entail, for instance, that a multicellular organism is at least as complex as its constituent cells. It may be useful and is intriguing to quantify and treat in a formal way the comparison of hierarchical complexity of various biological systems. Considering, for example, a group of free-living non-interacting bacteria, we would like to formally compare its complexity to that of the filamentous cyanobacterium *Anabaena catenula* in which some cells differentiate into nitrogen-fixing heterocysts [**MW72, Lin68**], or to compare these to single-celled eukaryotic cells with their many organelles — some of which are believed to be of endosymbiotic origin [**Mar81**] — and then compare the complexity of these to that of the development and inversion in colonial spheres of *Volvox carteri* or the morphogenesis and differentiation of the stalk and spores of the cellular slime mold *Dictyostelium discoidium* from free-living single cells, and then to compare these to cellular communities that comprise higher metazoan individuals [**Bus87, Bon88, Gil94**]. Our goal is to offer a formal, quantitative way to do this by taking advantage of tools from algebra.

2. Information & Hierarchy in the Major Evolutionary Transitions

J. Maynard Smith and E. Szathmáry [**MSS95**] following L. Buss [**Bus87**] identify major transitions in evolution from replicating molecules to populations of molecules in compartments; from independent replicators to chromosomes; from RNA as gene and enzyme to DNA and protein (genetic code); from prokaryotes to eukaryotes; from asexual clones to sexual populations; from protists to animals, plants, fungi (multicellularity, cell differentiation); from solitary individuals to colonies (with non-reproductive castes); from primate societies to human society (language).

A main source of major transitions of evolution [**MSS95**] and those in the evolution of individuality [**Bus87**] on our planet involves change in the manner in which information is represented or interpreted. This suggests that the application of automata theory to the study of evolutionary transitions is appropriate. It is worth noting that the current models of genomic control over cell state in cellular and molecular biology are essentially automata models, as made explicit for example by Kauffman [**Kau72**] for feedback control of protein biosynthesis; by Krohn, Langer, and Rhodes [**KLR67**] for the Krebs cycle; and by Lindenmeyer [**Lin68**] for simple models of development. Taking the automata-theoretic viewpoint as a starting point, we apply Krohn-Rhodes algebraic automata theory [**KRT68, Eil76**] to gain insight into the mathematical consequences of such a viewpoint for biological systems.

A second major source of evolutionary transitions has been the appearance of new levels of hierarchical structure, as is obvious in the many examples discussed by [**MSS95**]. We axiomatize the notion of hierarchical complexity of automata models of biological systems, and then study the implications of the algebraic approach for the mathematical analysis of hierarchical complexity and its evolution.

3. Complexity Measures for Living Organisms

Attempts to measure the complexity of organisms in a systematic fashion have resulted in several proposals, surveyed by Bonner [**Bon88**, Ch. 6] and Raff and Kaufman [**RK83**, Ch. 11]. For example, the *genome size* — the number of base-pairs of DNA in the haploid complement of an organism's genome — has long been discussed as a measure of complexity (*cf.* Sparrow *et al* [**SPU72**], Cavalier-Smith [**Cav85**] and [**Bon88, RK83, MSS95**]), but varies widely even for closely related taxa of apparently similar complexity. Presumably the amount coding DNA, possibly omitting duplicated genes, would provide a better measure [**Cav85**].

The *number of cell types* [**Bon65, Kau69**] a multicellular organism may have during the course of its life cycle is an interesting complexity measure. Kauffman has argued that the number of cell types corresponds to the number of stable states for a given genome, relating this measure to an automata-theoretic viewpoint. Bonner has shown that the size of aclonal organisms correlates with internal division of labor (as measured by number of cell types) and studied the question of why increased size should require more cell types [**Bon88**]. Note that the number of cell types is a statistic whose value may well depend on our current biochemical knowledge and whose estimated value may be subject to variance of opinion.

Our approach here will be to take the informal notion *number of levels in a hierarchy of organization* as the motivation for axioms on hierarchical complexity measures for biological systems. The measure we shall define has the virtue of mathematical precision and well-definedness without the requirement of knowing the future of biological theories. It satisfies a set of axioms that any good measure of hierarchical complexity should. Thus, while its particular values on automata models of a biological system may be difficult to calculate or adequate models may not be available until further biological details unfold, the value of the complexity measure on each particular automata model of any particular biological system has an unchanging value. Furthermore, our hierarchical complexity measure is related by numerical dominance to all others satisfying the complexity axioms as the unique measure having this maximality property.

4. Axiomatizing Complexity

In our axiomatization of hierarchical complexity, we will not be concerned with how a biological system is built, or even about the 'right' components to identify or the 'right' boundaries to draw between its levels. Generally one expects that endeavoring to answer the above will require one to frame difficult and important questions of research for biochemistry, molecular genetics, cellular biology, and physiology, anatomy, development and ecology of particular organisms.

Fortunately, by asking only for the hierarchical complexity and not for an explicit hierarchical description, the above issues can be initially avoided and addressed as the state of our biological knowledge expands. Assuming the automata-theoretic modelling of biological systems can be done in principle is all

that is needed.

Given any automaton X, we canonically associate to it a transformation semigroup generated by considering the semigroup of state-transition mappings induced by all possible sequences of inputs to the automaton acting on its states. See the appendix for details. The appendix collects the mathematical notions and notation used in the sequel. Readers who are biologists may wish to skip most of the appendix on first reading. Mathematicians feel more comfortable to read through it now or refer to it as necessary. The appendix also explains and gives examples of the notion of wreath product of transformation semigroups, which formalizes the intuitive concept of hierarchical composition of automata.

4.1. Hierarchical Complexity Axioms. A function $c : \mathcal{A} \to \mathbb{N}$, from transformation semigroups canonically associated to finite automata (modelling organisms) to the natural numbers is called a *hierarchical complexity measure* if it satisfies:

1. *If X and Y are combined hierarchically, the result has complexity not exceeding the sum of complexities of X and Y.*
   (This corresponds to the wreath product of transformation semigroups.)
   $$c(X \wr Y) \leq c(X) + c(Y)$$
2. *If X and Y are combined, but do* not *interact, the complexity of the whole is just the maximum of complexities of X and Y.*
   (This corresponds to direct product for transformation semigroups, and to direct product or disjoint union of automata.)
   $$c(X \times Y) = \max(c(X), c(Y))$$
3. *If Y can do everything X can, then X is not more complex than Y.*
   $$\text{If } X \prec Y, \text{ then } c(X) \leq c(Y)$$
4. *Every X can be constructed from components of minimal complexity.*
   $$\text{For all } X,\ X \prec X_n \wr \cdots \wr X_1, \text{ for some } X_i \text{ with } c(X_i) \leq 1$$
5. *Initial Condition: If G is a finite group, then $c(G) \leq 1$. If A has only trivial subgroups then $c(A) = 0$.*

Axioms 1, 2, 3 should obviously be properties of any good hierarchical complexity measure. Axiom 4 reflects the scientist's (partially reductionist) optimism that everything should be constructible from fundamental building blocks, while axiom 5 says that (at least) certain elementary building blocks should have low complexity.[2]

If, in addition, it satisfies the following axiom, c is called *the [maximal] hierarchical complexity measure.*

[2]The somewhat technical looking axiom 5 is equivalent to the conjunction of two more transparent conditions: (5a) systems which cannot count modulo n for any $n \geq 2$ should have complexity 0; and (5b) those finite systems in which the effect of every input can be undone without affecting computational power have complexity at most 1. It is clear that the latter have almost no life-like properties, beyond a kind of crystalline homeostatic symmetry, as they maintain reversibility in the face of all inputs and thus cannot 'die'. The semigroups associated with them are the so-called finite right-simple semigroups, and have form $G \times B^r$, where G is a group and B^r is a right-zero semigroup (by definition, satisfying the law $xy = y$).

6. *Maximality: If $f : \mathcal{A} \to \mathbb{N}$ satisfies the preceding axioms, then for all X, we have $f(X) \leq c(X)$.*

5. Krohn-Rhodes Theory and the Complexity Measure

We now show that existence and uniqueness of the hierarchical complexity measure follows from the Krohn-Rhodes Theorem [**KR65**]:

THEOREM 5.1 (**Prime Decomposition Theorem**). *For every finite transformation semigroup X, there exist finite groups and semigroups X_i with no nontrivial subgroups such that*

$$X \prec X_n \wr \cdots \wr X_1. \qquad (\star)$$

(One can build any X from group and aperiodic semigroup automata.) Moreover, if G is a finite simple group which divides the semigroup of X, then G divides the semigroup of X_i for some $1 \leq i \leq n$.

THEOREM 5.2. *There exists a unique maximal hierarchical complexity measure $cpx : \mathcal{A} \to \mathbb{N}$. To define $cpx(X)$, consider all decompositions of the form $(\star)$. We have :*

$$X \prec A_n \wr G_n \wr A_{n-1} \wr \cdots \wr A_i \wr G_i \wr A_{i-1} \wr \cdots \wr A_1 \wr G_1 \wr A_0,$$

where A_i has no nontrivial subgroup and G_i is a nontrivial group.

Then $cpx(X) =$ least n for which such a decomposition of X exists.

Note: *$cpx(X)$ is the least number of active computing (group) levels required in order to build X hierarchically from simple components.*

Proof: Axioms 4 and 5 are immediate for cpx, and 3 follows from transitivity of '$\prec$'. Since the wreath product of groups is a group and the wreath product of aperiodics is aperiodic, axiom 1 follows by hierarchically combining minimal length decompositions of X and Y. By taking direct products of components in shortest decompositions of X and Y, one obtains axiom 2. Finally, if c is a complexity measure satisfying axioms 1-5, then taking a shortest Krohn-Rhodes decomposition of X as above:

$$\begin{aligned} c(X) &\leq c(A_n \wr G_n \wr A_{n-1} \wr \cdots \wr A_1 \wr G_1 \wr A_0) \text{ by axiom 3} \\ &\leq c(A_n) + c(G_n) + \cdots + c(G_1) + c(A_0) \text{ by axiom 1} \\ &\leq n \text{ by axiom 5} \\ &\leq cpx(X), \end{aligned}$$

so axiom 6 holds. Moreover, the maximal complexity measure is unique, for, if cpx' were another, then $cpx'(X) \leq cpx(X) \leq cpx'(X)$ by two applications of axiom 6. □

6. Continuity & Jumpiness in the Evolution of Complexity

Assuming descent is "as smooth as can be" — without 'hopeful monsters" and moreover without any intermediates being skipped over — in the evolution of hierarchical complexity,[3] we ask, *How does complexity increase, smoothly or with jumps?*

[3] Without the assumption of smooth evolution, the jumps cannot be bounded unless some other constraints are accepted.

Formally, suppose that during descent with modification we have a sequence of organisms (represented by the semigroups associated to the finite-state automata modelling them) of increasing computational power:

$$S_1 \prec S_2 \prec \cdots \prec S_n.$$

Suppose, furthermore, that no proper intermediates could possibly be introduced into the sequence:

$$\text{If } S_i \prec T \prec S_{i+1}, \text{ then } S_i \cong T \text{ or } T \cong S_{i+1}.$$

Each division $S_i \prec S_{i+1}$ is "maximal". A division $S_i \prec S_{i+1}$ is called *proper* if $S_i \neq S_{i+1}$. We will study the hierarchical complexity changes possible in a single step of such an unrefinable chain.[4]

7. Bounds on Complexity Jumps in Evolution

[This section assumes a knowledge of semigroup theory and algebraic automata theory such as may be obtained from [**KRT68, Eil76**] or the survey section of [**AHNR95**]. The appendix gives definitions of terms and notation used here.]

THEOREM 7.1 (**Jump Lemma**). *Let $S \prec T$ be a maximal proper division of finite semigroups. Then either:*
(1) *S is a homomorphic image of T and*

$$cpx(S) \leq cpx(T) \leq cpx(S) + 1,$$

or
(2) *S is a maximal subsemigroup of T and*

$$cpx(S) \leq cpx(T) \leq 2\,cpx(S) + 1.$$

Proof: By hierarchical complexity axiom 3, one has that $cpx(S) \leq cpx(T)$. By definition of division ($\prec$), we have that $S \twoheadleftarrow T' \leq T$ for some subsemigroup T' of T mapping homomorphically onto S. By maximality of the division, it follows that either (1) $T \cong T'$ (so $S \twoheadleftarrow T$ is a maximal proper surmorphism [**Rho67, KRT68**]) or (2) $S \cong T'$ (so S is a maximal proper subsemigroup of T [**GGR68, KRT68**]).

For case (1), by the classification of maximal proper surmorphisms ([**Rho67**], [**KRT68**] or [**RW89**]), the surjective morphism φ is either (i) injective when restricted to each subgroup of T, hence by the Fundamental Lemma of Complexity [**Rho68, Rho71b, Rho74**] or [**Til76**, Ch. XII], $cpx(S) = cpx(T)$; or (ii) for regular elements $t_1, t_2 \in T$, if $\varphi(t_1) = \varphi(t_2)$ then t_1 and t_2 are $\mathcal{L}$-equivalent, hence by [**AHNR95**], [**KRT68**], or [**RW89**], $cpx(T) \leq cpx(S) + 1$.

In case (2), S is a maximal proper subsemigroup of T. And by [**GGR68**] or [**KRT68**], $T \setminus S \subseteq J$, for some $\mathcal{J}$-class J of T. Let I be the two-sided ideal

$$I = \{x \in T : J \not\leq_J x\}.$$

Now $T = I \cup S$, whence by "V-union-T" technique [**KRT68**, Ch. 5, Sec. 4], [**Eil76**, Prop. 3.5] or [**Neh95**, Sec. 3], we have $T \prec \overline{I}^{\bullet} \wr \overline{S}^{\bullet}$, and $cpx(T) \leq cpx(I) + cpx(S)$. Now $I = I \setminus J \cup J$. We have $I \setminus J \subset S$ and so $cpx(I \setminus J) \leq cpx(S)$. Let E be a

[4] One might also consider unrefinable chains with each step either increasing or decreasing in computational power. Since this just corresponds to the reversal of some maximal divisions '$\prec$', our results — which bound the complexity changes on the two sides of the '$\prec$' – will apply also to this more general situation.

system of idempotents for I (one idempotent from each maximal non-null $\mathcal{J}$-class). Then $cpx(EIE) = cpx(I)$ by the Reduction Theorem ([**Til76**, Ch. XII]), and

$$\begin{aligned} EIE &= E(I \setminus J \cup J)E \\ &= E(I \setminus J)E \cup EJE \\ &\subset I \setminus J \cup G_e, \end{aligned}$$

where G_e is the maximal subgroup of J containing an $e \in E$, or empty if J is a null $\mathcal{J}$-class.

We have $EIE^\bullet \prec \overline{I \setminus J}^\bullet \wr \overline{G_e}^\bullet$. Now $cpx(I) = cpx(EIE) \leq cpx(I \setminus J) + 1 \leq cpx(S) + 1$. Therefore

$$cpx(S) \leq cpx(T) \leq cpx(I) + cpx(S) \leq 2\, cpx(S) + 1.$$

□

To summarize: In case (1) (the maximal proper surmorphism case), $cpx(S) \in \{cpx(T), cpx(T) - 1\}$, whence hierarchical complexity increases by at most one: $cpx(S) \leq cpx(T) \leq cpx(S)+1$. While in case (2) (the maximal subsemigroup case), the bounds on complexity increase, assuming an unrefinable chain of descent, are given by

$$cpx(S) \leq cpx(T) \leq 2\, cpx(S) + 1.$$

Complexity may more than double!

8. Sharpness of the Bounds: 'Genius Jumps'

We considered a sequence of finite semigroups

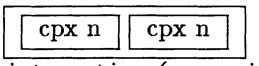
$$S_1 \prec S_2 \prec \cdots \prec S_n$$

representing an as continuous as possible sequence of organisms and showed that always

$$\boxed{cpx(S_i) \leq cpx(S_{i+1}) \leq 2\, cpx(S_i) + 1.}$$

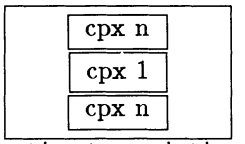
In fact, it is possible to construct $S_i \leq S_{i+1}$, a maximal proper subsemigroup as above with $cpx(S_{i+1}) = 2\, cpx(S_i)+1$, so these bounds are attained. The explicit construction of such a pair involves techniques beyond the scope of this paper, and full details will be published elsewhere [**NR**]. However, it is possible to give a sketch of the source of complexity increase in the examples so far constructed.

Beginning with a structure containing complexity n components, which do not interact or interact only weakly,



Two components not interacting (pre-existing structures).

a superstructure is constructed in which the complexity n components interact hierarchically, resulting in a chain with a new intermediate level,



New biological system integrating two existing subsystems hierarchically.

such that the biosystem cannot globally be described (in the sense of a giving a shortest Krohn-Rhodes decomposition) with less than $2n + 1$ essential (non-trivial group) coordinate levels.

9. Concluding Remarks

One observes that the 'genius jumps' whose structure is outlined above rely on existing fragmentation/inefficiency or they would not be possible. Two major factors in the major evolutionary transitions [**Ohn70, Bus87, MSS95**] discussed above are (1) change in *information* encoding/use and (2) *hierarchical structuring* of pre-existing units (with adaptations): division of labor / specialization / differentiation, duplication and divergence, symbiosis, epigenesis. The structure we observed in the above-sketched 'genius jumps' is very close conceptually to transitions of type (2) and may be typical of evolutionary complexity jumps. We see that possibly divergent duplicated units or, more generally, duplicate complexity units are being integrated to yield higher level complexity in the mathematical examples and in case (2). This mechanism is evident in symbiogenesis, epigenesis, and the rest of the case (2) examples. The mathematical and biological examples rely on fragmentation or unstructured duplication, and eliminate it, to achieve a 'genius jump'. Whether and how shifts in the way biological systems use information, *i.e.* type (1) transitions, can be understood in this manner remains to be explored.

Evolution of complexity and evolution of development in living systems are not well enough understood, but a mathematical treatment of hierarchical complexity may shed some light on these problems. Current controversies of punctationalism vs. gradualism in evolution may seem naïve in the pure light of algebra once a deeper mechanistic understanding of genetic control in development and its interplay with extra-genomic factors has been attained. If one accepts that organisms may be modelled as finite automata, then hierarchical complexity increase may be better understood via the algebraic theory of automata. Indeed, our results show that large jumps in hierarchical complexity are possible in principle within the strict bounds proved here and suggest that analogous abstract mathematical structure may lie behind many or all such evolutionary complexity jumps.

It is only fair to mention that the notion of hierarchical combination we considered in defining the complexity of an automata model of an organism was restricted in the sense that information about states could never flow back to higher levels

in the hierarchy. One could remove this restriction by developing an alternate approach using the *block product* of semigroups (*e.g.* [**RW89**]) rather than wreath products of transformation semigroups. The analog of the Prime Decomposition Theorem is available for the block product, but the analysis of complexity is complicated by the fact that this product is not associative on the class of all finite semigroups.

Appendix: Mathematical Concepts and Notation

This appendix collects some standard mathematical notation for algebraic automata and semigroup theory.

Semigroups, Automata, Transformation Semigroups. A *semigroup* is a set S with a binary multiplication defined on it satisfying the associative law: $(xy)z = x(yz)$. An *automaton* $X = (Q, S)$ is a set of states Q together with, for each symbol s of an input alphabet S, an associated [state-transition] function from Q to Q (also denoted by s). A special case of this is a *transformation semigroup* (Q, S), which is a right action of a semigroup S on a set of states Q, *i.e.*, a rule taking $q \in Q$ and $s \in S$ to $q \cdot s \in Q$, satisfying $(q \cdot s) \cdot s' = q \cdot ss'$ for all $q \in Q$, and $s, s' \in S$. Given any automaton X we canonically associate to it a transformation semigroup generated by considering the semigroup of state-transition mappings induced by all possible sequences of inputs to the automaton acting on its states.[5] To avoid trivialities, we shall always assume that state sets, input sets, and semigroups are non-empty. Generally we shall require that two distinct inputs $a \neq a'$ to an automaton differ at least at one state x: $x \cdot a \neq x \cdot a'$. Such an automaton is called *faithful*. The restriction of faithfulness is really not essential, as one can 'faithfulize' X by identifying input symbols which have the same action.

For semigroups, a *homomorphism* (or, more briefly, a *morphism*) from S to T is a function $\varphi : S \to T$ such that $\varphi(ss') = \varphi(s)\varphi(s')$ for all $s, s' \in S$. A morphism known to be surjective (onto) will be called a *surmorphism* and be denoted with a double-headed arrow $\varphi : S \twoheadrightarrow T$; the semigroup T is then said to be a *homomorphic image* of S. An injective (one-to-one) morphism φ is called an *embedding* of S into T. T is a *subsemigroup* of S if T is a subset of S closed under multiplication. We write $T \leq S$ if T is a subsemigroup of S or, more generally, embeds in S. If there is a bijective (one-to-one and onto) morphism from S to T, we say that S and T are *isomorphic* and write $S \cong T$. The notation $S \prec T$ means S is a homomorphic image of a subsemigroup of T. Intuitively, T can emulate any computation of S. In such a case, we say S *divides* T.

If $X = (Q, S)$ and $Y = (U, T)$ are automata or transformation semigroups a *morphism* φ from X to Y consists a function $\varphi_1 : Q \to U$ and a mapping $\varphi_2 : S \to T$ (which is required to a semigroup morphism in the transformation semigroup case), such that for all $q \in Q$ and $s \in S$, one has $\varphi_1(q \cdot s) = \varphi_1(q) \cdot \varphi_2(s)$. Surmorphism, embedding, isomorphism and division of transformation semigroups are defined analogously. If Y divides X, it is also common to say that X *covers* Y

[5] It is also possible to consider automata with output behavior. Under standard conventions, output is a function of the current state or of the current state and input symbol. Knowing the state (and possibly input symbol) of an automaton allows one to recover output behavior as necessary, so it need not be considered explicitly.

and that X *emulates* Y.

New Automata from Old. For transformation semigroups $X = (Q, S)$ and $Y = (U, T)$, their *wreath product* $Y \wr X$ is the transformation semigroup with states $U \times Q$ and semigroup consisting of all pairs (f, s) where $f : Q \to T$ and $s \in S$ with action: $(u, q) \cdot (f, s) = (u \cdot f(q), q \cdot s)$. Thus the action of input (f, s) on the X component is independent of the Y component but depends only on the input, while the action on Y depends on the input and the state of the X component. We have multiplication $(f', s')(f, s) = (h, s's)$ where $h(q) = f'(q)f(q \cdot s)$, a product in T. To understand action and the multiplication in the semigroup, observe that

$$((u, q)\cdot(f', s'))\cdot(f, s) = (u\cdot f'(q), q\cdot s')\cdot(f, s) = (u\cdot f'(q)f(q\cdot s), q\cdot s's) = (u\cdot h(q), q\cdot s's),$$

and indeed, the element $h(q) \in T$ applied to u is a function only of the input and the state q, while the element $s's \in S$ applied to q depends only on the input $(f', s')(f, s)$. The wreath product is an associative product on the class of all transformation semigroups. Notice that there is a hierarchical dependence of Y and on X. By iterating this construction, one may take the wreath product of any sequence of components.

The *direct product* $(U, T) \times (Q, S)$ of automata [resp. transformation semigroups] has state set $U \times Q$ and inputs [resp. semigroup] $T \times S$ with componentwise action: $(u, q) \cdot (t, s) = (u \cdot t, q \cdot s)$. This is the case of no interaction between components.

A *cascade* of automata X and Y as above has states $U \times Q$ and any set of inputs (f, s) with $f : Q \to T$ and $s \in S$, acting on the states as for the wreath product. The wreath product is the transformation semigroup version of a generic cascade of automata: It is easy to see that every cascade of two automata (including of course their direct product) embeds in the wreath product of their associated transformation semigroups (*cf.* [**Eil76, KRT68**]). Thus, the wreath product provides the formal algebraic framework for the notion of hierarchical combination of automata.

Given a semigroup S, we define $S^\bullet$ to be S if S contains an identity element $1 \in S$ with $s1 = 1s = s$ for all $s \in S$, otherwise we take $S^\bullet$ to be S with new identity element 1 adjoined. If $X = (Q, S)$ is an automaton or transformation semigroup, we define $X^\bullet = (Q, S^\bullet)$, where the identity of $S^\bullet$ acts as the identity function on all states in Q. Also $\overline{X} = (Q, \overline{S})$, where for each state $q \in Q$ a constant map taking value q has been adjoined as an element of S.

The *disjoint union* of automata $X \sqcup Y$ has state set $Q \times U$ and inputs $S \sqcup T$, with

$$(q, u) \cdot i = \begin{cases} (q \cdot i, u) & \text{if } i \in S \\ (q, u \cdot i) & \text{if } i \in T \end{cases}$$

Observe that this automaton generates the direct product at the associated transformation semigroup level if X and Y contain identity transformations. Whence, $X \sqcup Y$ embeds in $X^\bullet \times Y^\bullet$.

For a semigroup S, one obtains a canonically its so-called *right regular representation* $(S^\bullet, S)$, with states $S^\bullet$, semigroup S and action $s \cdot s' = ss'$ for all $s \in S^\bullet$ and $s' \in S$. *If S and T are semigroups, we shall write $S \wr T$ for the wreath product of their right regular representations.*

Ideals, Groups, and Green's Relations in Semigroups. A subsemigroup I of S is called an *ideal* if $SIS \subseteq I$, and a *left ideal* if $SI \subseteq I$. *Right ideals* are defined

analogously. One says that for $s, t \in S$, that $s \geq_J t$ (read: "s is $\mathcal{J}$-*above*" t) if $t \in S^1 s S^1$, the principal ideal generated by s. This comprises a transitive relation on S. One says that t is $\mathcal{J}$-*equivalent* to s if s and t generate the same principal ideals. A $\mathcal{J}$-*class* of S is maximal subset of S consisting of $\mathcal{J}$-equivalent elements. The notation $s >_J t$ (read: "s is strictly $\mathcal{J}$-above t") means that s is $\mathcal{J}$-above but not $\mathcal{J}$-equivalent t, while $s \not>_J t$ is the negation of this. A $\mathcal{J}$-class is said to be *null* if $ss' \notin J$ for all $s, s' \in J$. An element $s \in S$ is *regular* if its $\mathcal{J}$-class is not null.

By considering left principal ideals, one obtains analogously the transitive $\mathcal{L}$-*relation*: $s \geq_L t$ (read: "s is $\mathcal{L}$-above t") if $t \in S^1 s$. So one has also notions of $\mathcal{L}$-*equivalence* and $\mathcal{L}$-*class*. The $\mathcal{R}$-*relation* $\geq_R$ may be defined as the dual of the $\mathcal{L}$-relation.

A *group* is a semigroup S with exactly one $\mathcal{L}$-class and exactly one $\mathcal{R}$-class; or, equivalently, S has an identity e and for each element s in S, there exists an *inverse* s' in S with $ss' = s's = e$. A group is called *simple* if its homomorphic images are just itself and the one element group (up to isomorphism). A finite semigroup each of whose subgroups have only one element is called *aperiodic*.

Examples of Hierarchical Combination. It may be useful to note some familiar examples of wreath product coordinates:

Symmetries on a Pentagon. Let us imagine a regular polygon with n vertices in the plane with one vertex topmost. Let $\mathbb{Z}_n$ denote the cyclic group of order n. By our convention $\mathbb{Z}_n \wr \mathbb{Z}_2$ denotes the wreath product of the right regular representations of $\mathbb{Z}_n$ and $\mathbb{Z}_2$. Symmetries of a pentagon comprise the dihedral group $\mathbf{D}_5$ (resp. $\mathbf{D}_n$ for the regular n-gon), which is covered by such a wreath product. If we paint one side of the pentagon white and the other black, and number the vertices of the pentagon clockwise on the white side from 0 to 4, then a pair $(k, color)$ determines a configuration of the pentagon where we understand $k \in \mathbb{Z}_5$ as the number on the currently topmost vertex and $color \in \mathbb{Z}_2$ as the color of the face we see, either 0 denoting white or 1 denoting black. This gives a coordinate representation of the state of the pentagon.

As for symmetries, let us consider first the 'flip' F — rotation of 180 degrees about the axis through the center of the figure and topmost vertex. The effect of this action is clearly $(k, color) \cdot F = (k + 0, color + 1)$, since the same vertex is still topmost but the other side now faces us. Let us consider the rotation R counterclockwise through $\frac{360}{n}$, or 72 degrees for the pentagon. Rotation R leaves the color component unchanged, but, if the color is white, increases the topmost vertex number by 1, whereas if the color facing us is black, the topmost vertex number is decreased by 1 (modulo 5).

Thus we have $(k, color) \cdot R = (k + f(color), color + 0)$ where $f(white) = 1$ and $f(black) = 4$ (that is, -1 modulo 5). Since 0 is in the semigroup $\mathbb{Z}_2$ and f is a function from the state in $\mathbb{Z}_2$ to the semigroup $\mathbb{Z}_5$, it is clear that this expresses R as a member of the wreath product.

It is well-known that F and R generate all symmetries of the polygon (with FRF the inverse of R). Since the semigroup component of a wreath product is closed under composition, all symmetries are expressible in this hierarchical coordinate manner. This coordinatization constitutes a covering of the symmetry automaton $(pentagon\ states, \mathbf{D}_5)$ by $\mathbb{Z}_5 \wr \mathbb{Z}_2$. Actually in this case, we even have an embedding, since states and symmetries have unique representatives ('lifts') in

the wreath product. Similarly, replacing 5 with any integer $n \geq 3$ in the argument above yields a wreath product coordinatization in $\mathbb{Z}_n \wr \mathbb{Z}_2$ of the symmetry group $\mathbf{D}_n$ of a regular n-sided polygon.

Decimal Expansion and Clocks. The standard clock face combines a modulo 60 counter, $\mathbb{Z}_{60}$, for minutes (and another for seconds) with a modulo 12 counter, $\mathbb{Z}_{12}$, for hours. The top of the hierarchy is seconds, on which depend minutes, and then hours, constituting a convenient wreath product embedding of a modulo $43,200$ $(60 \times 60 \times 12)$ counter for seconds in the half-day. Passage of one second adds 1 to the seconds counter and zero elsewhere, unless the seconds state is 59 in which case the next (minute) automaton adds 1 to its state and zero is added to the hours counter; unless both minutes and seconds have state 59, in which case, we also get plus 1 on hours. Thus, the passage of one second is described in these familiar wreath product coordinates. Consideration of `a.m.` or `p.m.` gives us another $\mathbb{Z}_2$ counter at a deeper level of dependency.

The reader is invited to check that the addition and carry rules for base 10 addition that she or he learned in elementary school are in fact 'just' a wreath product coordinatization of the right regular representation of the additive semigroup of natural numbers $(\mathbb{N}, +)$ by a cascade of (countably many) modulo ten counters $\mathbb{Z}_{10}$. In the case of the non-negative real numbers under addition, the decimal expansion also induces a covering (by a $\mathbb{Z}$-indexed hierarchy of $\mathbb{Z}_{10}$'s), but note that this wreath product coordinatization is no longer an embedding since some numbers have more than one decimal representation.

Acknowledgements

The first author wishes to thank Professor George Estabrook of the University of Michigan for encouragement and some valuable comments.

References

[AHNR95] B. Austin, K. Henckell, C. Nehaniv and J. Rhodes, Complexity and Subsemigroups via the Presentation Lemma, *Journal of Pure & Applied Algebra*, Vol. 101, No. 3, pp. 245–289, 1995.

[Bon65] J. T. Bonner, *Size and Cycle*, Princeton University Press, 1965.

[Bon88] J. T. Bonner. *The Evolution of Complexity by Means of Natural Selection*, Princeton, 1988.

[Bus87] L. W. Buss. *The Evolution of Individuality*, Princeton, 1987.

[Cav85] T. Cavalier-Smith. *The Evolution of Genome Size*, Wiley, 1985.

[DeR96] E. M. DeRobetis, Homeotic genes and the evolution of body plans. In C. R. Marshall and J. Wm. Schopf, eds., *Evolution and the Molecular Revolution*, Jones and Bartlett Publishers, 1996.

[Eil76] S. Eilenberg. *Automata, Languages and Machines*, volume B. Academic Press, New York, 1976.

[Gil94] S. F. Gilbert. *Developmental Biology*, 4th edition, Sinauer, 1994.

[GGR68] N. Graham, R. Graham, and J. Rhodes, Maximal subsemigroups of finite semigroups, *Journal of Combinatorial Theory*, Vol. 4, No. 3, pp. 203–209, 1968.

[Kau69] S. A. Kauffman, Metabolic stability and epigenesis in randomly constructed genetic nets. *Journal of Theoretical Biology*, **22**, 437–467, 1969.

[Kau72] S. A. Kauffman, Organization of Cellular Genetic Control Systems. In J. D. Cowan, ed., *Some Mathematical Questions in Biology II*, Lectures on Mathematics in the Life Sciences, Vol. 3, American Mathematical Society, pp. 62–116, 1972.

[KR65] K. Krohn and J. Rhodes, Algebraic theory of machines, I. Prime decomposition theorem for finite semigroups and machines, *Transactions of the American Mathematical Society*, **116**, 450–464, 1965.

[KLR67] K. Krohn, R. Langer and J. Rhodes, Algebraic Principles for the Analysis of a Biochemical System. *Journal of Computer & Systems Sciences*, **1**, 119–136, 1967.

[KRT68] K. B. Krohn, J. L. Rhodes and B. R. Tilson. The Prime Decomposition Theorem of the Algebraic Theory of Machines (Chapter 5), Local Structure Theorems for Finite Semigroups (Chapter 7), and Homomorphisms and Semilocal Theory (Chapter 8). In M. Arbib, ed., *Algebraic Theory of Machines, Languages, and Semigroups*, Academic Press, 1968.

[Lin68] A. Lindenmeyer, Mathematical Models for Cellular Interactions in Development: I. Filaments with One-sided Inputs, & II. Simple and Branching Filaments with Two-sided Inputs, *Journal of Theoretical Biology*, **18**, 290–299 and 300–315, 1968.

[MSS95] J. Maynard Smith and E. Szathmáry. *The Major Transitions in Evolution*, W. H. Freeman, 1995.

[Mar81] L. Margulis, *Symbiosis in Cell Evolution*, Freeman, 1981.

[MW72] G. J. Michison and M. Wilcox, Rules governing cell division in *Anabaena*, *Nature*, **239**, 110-111, 1972.

[Neh95] C. L. Nehaniv, Cascade decomposition of arbitrary semigroups. In J. Fountain, ed., *Semigroups, Formal Languages and Groups*, (NATO Advanced Study Institute, University of York, 7-21 August 1993), Kluwer Academic Publishers, pp. 391–425, 1995.

[Neh96] C. L. Nehaniv, Complexity of finite aperiodic semigroups and star-free languages. In J. Almeida, G. M. S. Gomes, P. V. Silva, eds., *Semigroups, Automata and Languages* (University of Porto, 20-24 June 1994), World Scientific, pp. 195–209, 1996.

[NR] C. L. Nehaniv and J. L. Rhodes, The evolution of biological complexity from an algebraic perspective, *forthcoming.*

[Ohn70] S. Ohno, *Evolution by Gene Duplication.* Springer-Verlag, 1970.

[RK83] R. A. Raff and T. C. Kaufman. *Embryos, Genes, and Evolution: The Developmental-Genetic Basis of Evolutionary Change.* Macmillan Publishing Co., 1983. Reprinted with new introduction and preface, Indiana University Press, 1991.

[Rho67] J. Rhodes, A homomorphism theorem for finite semigroups, *Mathematical Systems Theory*, **1**, 289–304, 1967.

[RW89] J. Rhodes and P. Weil, Decomposition techniques for finite semigroups, using categories I & II, *Journal of Pure & Applied Algebra*, **62**, 269–284 and 285–312, 1989.

[Rho68] J. Rhodes, The fundamental lemma of complexity for arbitrary finite semigroups, *Bulletin of the American Mathematical Society*, **68**, 1104–1109, 1968.

[Rho71a] J. L. Rhodes, *Applications of Automata Theory and Algebra via the Mathematical Theory of Complexity to Biology, Physics, Psychology, Philosophy, Games, and Codes.* University of California Library (unpublished book, 1971); revised edition forthcoming.

[Rho71b] J. Rhodes, Proof of the fundamental lemma of complexity (weak version) for arbitrary finite semigroups, *Journal of Combinatorial Theory, Series A*, **10**, 22–73, 1971.

[Rho74] J. Rhodes, Proof of the fundamental lemma of complexity (strong version) for arbitrary finite semigroups, *Journal of Combinatorial Theory, Series A*, Vol. 16, No. 2, pp. 209–214, 1974.

[RN] J. L. Rhodes and C. L. Nehaniv, Complexity theories in mathematics, *in preparation.*

[RT75] J. Rhodes and B. Tilson, A reduction theorem for complexity of finite semigroups, *Semigroup Forum*, **10**, 96–114, 1975.

[SPU72] A. H. Sparrow, H. J. Price, and A. G. Underbrink, A survey of DNA content per cell per chromosome of prokaryotic and eukaryotic organisms: Some evolutionary considerations. *Brookhaven Symp. Biol.* **23**, 451–494, 1972.

[Til76] B. Tilson, Depth Decomposition Theorem (Chapter XI) & Complexity of Semigroups and Morphisms (Chapter XII). In S. Eilenberg, *Automata, Languages, and Machines*, Vol. B, Academic Press, 1976.

SCHOOL OF COMPUTER SCIENCE & ENGINEERING, UNIVERSITY OF AIZU 985-80, JAPAN
E-mail address: nehaniv@u-aizu.ac.jp

DEPARTMENT OF MATHEMATICS, UNIVERSITY OF CALIFORNIA, BERKELEY, CA 94720, U.S.A.
E-mail address: rhodes@math.berkeley.edu

https://doi.org/10.1090/dimacs/037/03

DIMACS Series in Discrete Mathematics
and Theoretical Computer Science
Volume **37**, 1997

Inferring Evolutionary Trees from Polymorphic Characters, and an Analysis of the Indo-European Family of Languages

Maria Bonet, Cynthia Phillips, Tandy Warnow, and Shibu Yooseph

ABSTRACT. In this paper we consider the problem of constructing phylogenies in the presence of polymorphic character data. We also discuss the result of applying our techniques to studying the evolution of the Indo-European family of languages.

1. Introduction

Determining the evolutionary history of a set S of objects (taxa or species) is a problem with applications in a number of domains such as biology, comparative linguistics, and literature. Primary data used to compare different taxa (whether biological species, populations, or languages) can be described using *characters*. Characters describe aspects of the elements of S. Typically, a character c partitions the set S into equivalence classes and can thus be described as a function $c : S \rightarrow Z_r$, where $Z_r = \{0, 1, ..., r-1\}$. The value $c(s)$ is called the *character state* of c on s, where $s \in S$.

In this paper we describe tree construction when characters are permitted to have *more than one character state* on a given object. We call this the *polymorphism problem*. The definition of a character is thus extended to $c : S \rightarrow 2^Z - \{\emptyset\}$. A character which is permitted to have more than one state on a given object will be called a *polymorphic character*, and one which can have only one state for every object is referred to as a *monomorphic character*.

Polymorphism is a well-documented phenomenon in several areas like biology [**9**] and historical linguistics [**13**]. Our further discussions will be restricted to historical linguistics, where we are interested in studying the evolution of languages. The techniques discussed here can be used for some biological data sets as well. Other techniques for dealing with biological data sets are mentioned in [**3**].

Research partly supported by NSF grant number CCR-9403447.

This work was performed under U.S. Department of Energy contract number DE-AC04-76AL85000.

Research partly supported by an NSF National Young Investigator Award under contract ccr-9457800 and from an NSF grant in Linguistics.

Research partly supported by a Fellowship from the Institute for Research in Cognitive Science at the University of Pennsylvania and also by a Fellowship from the Program in Mathematics and Molecular Biology at the University of California at Berkeley, which is supported by the NSF under grant no. DMS-9406348.

A version of this paper appeared in *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing* (Philadelphia, PA, 1996).

The major contribution of this paper is a methodology for inferring perfect phylogenies from monomorphic and polymorphic characters. Recent work in historical linguistics [**13**] has shown that perfect phylogenies should be obtainable from properly selected and encoded linguistic characters. Algorithms for constructing perfect phylogenies from monomorphic characters were used in [**13**] to analyze the Indo-European family of languages, whose first-order subgrouping had been argued for decades without resolution. The methodology we propose here significantly extends the range of the data that can be analyzed in historical linguistics. We have applied this methodology to the data set studied by Warnow, Ringe, and Taylor. Detection and resolution of polymorphism led to a modification of their initially proposed phylogeny, which was based only on monomorphic characters. Our methodology and its results were presented at the Symposium on the Frontiers of Science at the National Academy of Sciences in November 1995.

The rest of the paper is organized as follows. In Section 2, we briefly discuss historical linguistics character data, and the evolutionary tree construction criterion used in historical linguistics. In Section 3, we discuss the causes of polymorphism in historical linguistics. In Section 4, we discuss the problem of inferring trees from polymorphic data sets and describe algorithms for tree construction. In Section 5.3, we discuss methods to deal with data sets containing both polymorphic and monomorphic data sets. Finally, in Section 6, we present our analysis of the Indo-European data set studied by [**13**]. This is an extended abstract and as such, the proofs have been omitted as they will appear in the journal version of this paper.

2. Historical linguistics

In historical linguistics, the following types of characters [**13**] occur: *Lexical characters*, where the character is a semantic slot (word meaning); *Morphological characters*, where the character is generally a grammatical feature; and *Phonological characters*, where the character is a sound change.

Once the raw character data is available, the encoding of the character states is done using cognation judgements based on the *Comparative Method* [**13**].

In [**13**], it was also shown that all properly selected and encoded characters should be *convex* (or *compatible*) on the true evolutionary tree. A character is convex on a tree T if it is possible to label the internal nodes of T so that each character state of that character arises exactly once. A tree, in which every character is convex, is called a *perfect phylogeny.*

3. Polymorphism in historical linguistics

When studying the Indo-European data set, it was discovered that some characters were polymorphic, for example more than one word is available in a particular semantic slot (consider *big* and *large*). Polymorphism arises due to several reasons. Convergence of meanings over time, borrowing of synonyms from other languages, and the inability of modern-day linguists to detect subtle differences of meaning in words from ancient languages, can all produce polymorphic characters. Some such cases, like English *little* and *small,* arise by the convergence of meanings over time; others, like American English *stone* and *rock* (to describe a small chunk of the substance that can be thrown), are instances of replacement in progress (*rock* is replacing *stone* in that basic meaning in America). Still others, like English *big* and *large,* result from the borrowing of synonyms from different sources (*large* having

been borrowed from French, while *big* appears to be of Scandinavian origin, and both have undergone some shift of meaning within English; the original Old English word survives only in the name *Mitchell*). In working with older stages of languages we are not always able to determine which of several words most usually expressed a basic meaning; for example, Vedic Sanskrit presents us with four common verbs meaning *tie,* and because the texts are poetic, we cannot say with certainty which word was used to describe archetypal acts of tying (without further connotations) in ordinary speech; in such cases we can do no better than list all of the possible alternatives.[**10**]

It can be shown that the different manifestations of polymorphism in linguistics each can be described by the conflation of two or more distinct linguistic characters. Often we are able to determine the precise number of monomorphic characters that have merged into the polymorphic character.

As mentioned before, in linguistics, it has been observed that monomorphic characters are *convex* on the true evolutionary tree.

DEFINITION 1. *Given a set S of taxa defined by a set C of characters ($|C| = k$), where each $c_j \in C$ is a function $c_j : S \to (2^Z - \{\emptyset\})$, let T be a tree which is leaf-labelled by the taxa in S and with each internal node v labelled with a vector from $(2^Z - \{\emptyset\})^k$ such that the value of $c_j(v)$ is given by the j^{th} component of this vector. A character (polymorphic or monomorphic) c is* **convex** *on T if for all $i \in Z$, the set $X_{c,i} = \{v \in V(T) : i \in c(v)\}$ is connected. T is a* **perfect phylogeny** *if every character is convex.*

For polymorphism caused by convergence of convex monomorphic characters, polymorphism can be considered a *separation* problem.

DEFINITION 2. *A polymorphic character c with r states is* **separated** *into characters $\alpha_1, \ldots, \alpha_l$ by a function $f : \{0, 1, \ldots, r-1\} \to \{1, \ldots, l\}$ where $\alpha_j^{-1}(i) = c^{-1}(i)$ if $f(i) = j$. Undetermined values of $\alpha_1, \ldots, \alpha_l$ are arbitrary. In particular* singletons *maintain the spirit of character c. That is, if $f(i) \neq j$ for any $i \in c(s)$ (species s does not contain any state mapped to character j), then let $\alpha_j^{-1}(\alpha_j(s)) = \{s\}$ (s has a unique state for α_j).*

3.1. Problem 1: Separation into l convex characters.

Input: Set S of taxa defined by set C of characters.

Question: Can we separate each character into at most l monomorphic characters, so that a perfect phylogeny exists for the derived set of monomorphic characters?

Due to inadequate historical evidence, input data may not reflect the actual degree of polymorphism. Separation may be necessary to obtain convexity even if all input characters appear monomorphic. For example, consider four languages with three characters: $A = (1, 2, 1), B = (1, 2, 2), C = (1, 2, 1), D = (1, 2, 2)$. Suppose the first two characters convolve (meanings merge) and linguists detect only one of these characters for each language. This polymorphic character appears monomorphic: $A = (1, 1), B = (1, 2), C = (2, 1), D = (2, 2)$. There is no perfect phylogeny for this set, but we can separate the first character into two such that there is a perfect phylogeny: $A = (1, a, 1), B = (1, b, 2), C = (c, 2, 1)$, and $D = (d, 2, 2)$. Because of lost information, we cannot completely determine the *inferred* characters α_i (hence the use of singletons).

DEFINITION 3. *A tree T which has polymorphic characters is said to have* **load** *l if for every character $c \in C$ and every $v \in V(T)$, $|c(v)| \leq l$.*

3.2. Problem 2: l-load perfect phylogeny.

Input:: Set S of taxa defined by set C of (possibly) polymorphic characters.
Question:: Does an l-load perfect phylogeny exist?

We now state the Theorem that shows that the two problems are in fact equivalent.

THEOREM 4. *Given a set of taxa defined by a set C of polymorphic characters, T is an l-load perfect phylogeny for C if and only if we can separate each polymorphic character into at most l monomorphic characters such that T is also a perfect phylogeny for the derived set C'.*

4. Inferring Perfect Phylogenies from Polymorphic Characters

When the maximum permissible load for each character is not given, the problem of inferring perfect phylogenies is best stated as a *minimum load* problem. This is addressed in Section 4.1. When the maximum permissible load for each character is given, we have two algorithms which can construct perfect phylogenies; both are efficient when the number of characters is small. These algorithms are presented in Section 5. When the character set includes a sufficient number of monomorphic characters, we have a third algorithm which combines techniques for monomorphic and polymorphic characters. This algorithm is presented in 5.3.

4.1. Min Load Problems. When convexity of the monomorphic constituents of the polymorphic characters is a reasonable request, we may seek a tree with a pre-specified load bound, or else we may seek a tree with a minimum possible load bound. We call the latter problem the Minimum Load Problem.

We note that the Minimum Load Problem is NP-hard, since the question of whether a 1-load Perfect Phylogeny exists is NP-Complete [**2, 12**]. The 2-load Perfect Phylogeny Problem is the next question to consider. The various parameters to the problem are n, the number of species; k, the number of (polymorphic) characters; and r, the maximum number of states per character.

THEOREM 5.
(i): *The Min Load Problem can be solved in polynomial time for all fixed n.*
(ii): *The Min Load Problem can be solved in polynomial time when $r = 2$.*
(iii): *The Min Load Problem is NP-hard for all fixed k.*
(iv): *The Min Load Problem is NP-hard for all fixed $r \geq 3$.*
(v): *Determining whether a 2-load perfect phylogeny exists is solvable in polynomial time for all fixed n.*
(vi): *Determining whether a 2-load perfect phylogeny exists is solvable in polynomial time for $r = 2$.*
(vii): *Determining whether a 2-load perfect phylogeny exists is solvable in polynomial time for all fixed k.*
(viii): *Determining whether a 2-load perfect phylogeny exists is NP-complete for all fixed $r \geq 3$.*

This theorem shows that *any* polynomial time algorithm *requires* both k and l bounded (under $P \neq NP$ assumption). We note here that if r is bounded then l is

also bounded. Thus the case of bounded k and r is reduced to solving the case of bounded k and l.

5. Algorithms for Perfect Phylogenies from Polymorphic Characters

In this section we present the two algorithms for inferring perfect phylogenies from polymorphic data when we know the load bound. Although the algorithms we will present assume a universal load bound, these algorithms can be easily modified to allow individual load bounds for each character, and will achieve comparable running times. For the sake of clarity, we will present these algorithms as though the load bound is the same for each character; the runtimes of these algorithms when implemented to handle variable constraints are given within their respective sections.

5.1. A Combinatorial Algorithm for fixed k and l. The algorithm we present is an extension and simplification of the algorithm of Agarwala and Fernández-Baca [1]. For the remainder of this section the term *perfect phylogeny* refers to an l-load perfect phylogeny.

Because each character has only r states and each node can choose at most l of these in an l-load perfect phylogeny, the number of possible labels for nodes in the tree is $O(r^{lk})$. Let us call this set S^*, and note that $S \subseteq S^*$ (since otherwise some node in S has load greater than l). In contrast to the algorithm in [1], we do not require that the internal nodes be labelled distinctly from the species in S, and instead will permit species in S to be internal nodes because we can transform any perfect phylogeny in which some species in S label internal nodes into a perfect phylogeny in which all species label leaves by attaching a leaf for s to the internal node labelled by s.

We need some preliminary definitions and facts.

DEFINITION 6. *The* **Extended Hamming distance** *of* $e = (x, y)$ *is* $\sum_{c \in C} |c(x) \triangle c(y)|$, *where* $\triangle$ *denotes the symmetric difference. However, we will call this the* Hamming distance, *understanding this to refer to the extended Hamming distance.*

We note that if a perfect phylogeny exists for S, then one exists where the Hamming distance on any edge is exactly one. We will seek a perfect phylogeny with this property. Working with such perfect phylogenies allows us to quickly solve subproblems, because it limits the number of ways a (maximally refined) perfect phylogeny can be constructed.

DEFINITION 7. *(See* [6]*) Given* $x \in S^*$, *the equivalence relation* E_x *is the transitive closure of the following relation* E'_x *on* $S - \{x\}$: $aE'_x b$ *if there exists character* c *such that* $(c(a) \cap c(b)) - c(x) \neq \emptyset$. *We denote this set of equivalence classes by* $(S - \{x\})/x$.

Some facts follow from this definition. Let x be an internal node of a perfect phylogeny T.

Fact 1: Two species in S which are in the same equivalence class of $(S-\{x\})/x$ must be in the same component of $T - \{x\}$.

Fact 2: If a perfect phylogeny exists for $S \cup \{x\}$, then there is a perfect phylogeny T in which the components of $T - \{x\}$ have leaf sets which are the components of $(S - \{x\})/x$.

Fact 2 does not necessarily hold simultaneously for all internal nodes of a perfect phylogeny T. Instead the following fact is true for every internal node of T.

Fact 3: Consider T as rooted at x. Let G be an equivalence class of $(S-\{x\})/x$, and let $y = lca_T(G)$. Let v be a node of T on the path from x and y (thus $v = x$ or $v = y$ is also possible). Then there exist $H_1, \ldots, H_t$ in $(S - \{v\})/v$ such that $H_1 \cup \cdots \cup H_t = G$.

We now present a dynamic programming algorithm for constructing perfect phylogenies from polymorphic data. We define the *search graph* $SG = (V, E)$ as follows. Each vertex in V is associated with a pair $[G, x]$, where $G = S$ or $G \in (S-\{x\})/x$, and represents the question: *Does* $G \cup \{x\}$ *have a perfect phylogeny?".* The edges of the search graph are of the form $([G, x][S, x])$, and all pairs of the form $([G_1, x_1], [G_2, x_2])$ where $G_1 \subseteq G_2$ and x_1 and x_2 satisfy $\sum_{c \in C} |c(x_1) \triangle c(x_2)| = 1$. There are $O(r^{lk})$ nodes of type $[S, x]$, and $O(nr^{lk})$ of type $[G, x]$ (because there are at most n equivalence classes in $(S - \{x\})/x$). Also, there are $O(nr^{lk})$ edges of type $([G, x][S, x])$, and $O(nlkr^{lk+1})$ of type $([G_1, x_1], [G_2, x_2])$, since the outdegree of every node is at most lkr.

DEFINITION 8. *Given a node* $[G, x]$, *a set of nodes* $[H_1, y], [H_2, y], \ldots, [H_p, y]$ *such that (a) Hamming(x,y)=1 and (b)* $\cup_i H_i = G$ *is called a* **bundle.**

There can be multiple bundles going into $[G, x]$, corresponding to the maximally refined perfect phylogenies of $G \cup \{x\}$. If $[H_1, y], [H_2, y], \ldots, [H_p, y]$ is a bundle for $[G, x]$ and all the subproblems have perfect phylogenies, then there is a perfect phylogeny for $G \cup \{x\}$ with subtrees T_i labelled by H_i. We can also have a bundle of just one edge (i.e. ([G,y],[G,x])); such a bundle indicates the existence of a perfect phylogeny T for $G \cup \{y\}$ in which the node corresponding to y has only one child. This is necessary if we require all edges to have Hamming distance 1.

5.1.1. *The Algorithm PHYLOGENY(S).* First create the search graph G_S. For each node $[G, x]$, determine its bundles. Note that some incoming edges $([G_1, x_1], [G, x])$ may not correspond to any bundle because $(S - \{x_1\})/x_1$ does not have the proper form (i.e. G may not be the union of a subset of the components of $(S - \{x_1\})/x_1$). Remove such edges. Now for each bundle, compute the size of the bundle (number of edges) b_i and set a counter $\mathbf{count}_i$ equal to b_i. Each node $[G_1, x_1]$ that is a predecessor of node $[G_2, x_2]$ is given a pointer to the counter for its bundle. We initialize a queue of "true" nodes as empty.

We locate each node $[G, x]$ with $|G| = 1$, mark it as "true", and place it in the queue. We then pull a node $[G_1, x_1]$ out of the queue and process it as follows. For each edge in the search graph $([G_1, x_1], [G_2, x_2])$, we decrement the counter for the appropriate bundle into $[G_2, x_2]$. If the counter is decremented to 0, then all edges of the bundle have been set to true and node $[G_2, x_2]$ is added to the queue. When we have processed all edges out of node $[G_1, x_1]$ we choose another node from the queue and continue. If we ever try to enqueue a node of the form $[S, x]$, then the instance has a perfect phylogeny. If the queue is emptied without ever labelling a node of this form as "true", then there is no perfect phylogeny.

As we enqueue "true" nodes, we build a topology for a perfect phylogeny for the subproblem represented by that node, ultimately building one for the whole problem if it exists. We denote the topology of the perfect phylogeny for $[G, x]$ by $T[G, x]$. We enqueue $[G, x]$ when a bundle $[H_1, y], [H_2, y], \ldots, [H_p, y]$ is found such that each $[H_i, y]$ has been determined to be "true," and hence a topology $T[H_i, y]$ for each subproblem has already been determined. We create a new node v. If

$x \in S$, then we label the node x. Otherwise it remains unlabelled for now. A method for labelling these nodes is part of the proof of Theorem 9 (which we have omitted here). We take each of the trees $T[H_1, y], T[H_2, y], \ldots, T[H_p, y]$, merge the roots into a single node, and make this node a child of node v. Once $[G, x]$ has been enqueued, we construct the tree $T[G, x]$ and we do not consider any more edges entering $[G, x]$. Thus we only compute one topology per "true" subproblem.

LEMMA 1. *If there exists a perfect phylogeny for $S \cup \{x\}$, then the algorithm PHYLOGENY assigns true to $[G, x]$, for each $G \in (S - \{x\})/x$.*

THEOREM 9. *The algorithm PHYLOGENY(S) runs in time $O(r^{lk+1}lkn)$, and returns "yes" if and only if S has a perfect phylogeny.*

Comment: When individual load bounds l_c are given, the algorithm can be modified to run in $O(r^{L+1}Ln)$, where $L = \sum_{c \in C} l_c$.

5.2. A Graph-Theoretic Algorithm for Fixed k and l. In this section we give a graph-theoretic algorithm for the l-load perfect phylogeny problem. The algorithm we present is based upon a characterization of intersection graphs derived from l-load perfect phylogenies as a particular kind of vertex-colored triangulated (i.e. chordal) graphs. Based upon this characterization we will derive an efficient algorithm for the l-load perfect phylogeny problem when we can fix both l and k.

5.2.1. *Preliminary Definitions.* Let $G = (V, E)$ be a graph. A **vertex coloring** of G is a function $color : V \to Z$. We do not require that *color* be a *proper* coloring (a coloring function is proper if and only if $\forall (v, u) \in E$, $color(v) \neq color(u)$).

The **neighbour set** $\Gamma(v)$ of a vertex v is the set of all vertices in the graph adjacent to v. A vertex v is **simplicial** if $\Gamma(v)$ is a clique, i.e. all vertices in $\Gamma(v)$ are adjacent.

Given a graph $G = (V, E)$ and a vertex coloring $c : V \to Z$, a **monochromatic clique** in G is a clique with vertex set $V_0 \subset V$ such that $color(v) = color(w)$ for all $v, w \in V_0$. A graph $G = (V, E)$ is **triangulated** if it has no induced cycles of size four or greater. Given a vertex-colored graph $G = (V, E)$, we say that G is **l-triangulated** if G is both triangulated and has no monochromatic cliques of size greater than l. We say that G has an **l-triangulation** $G' = (V, E')$ if $E \subseteq E'$ and G' is l-triangulated.

Let $I = (S, C)$ be an input to the phylogeny problem. For $\alpha \in C$, we define $\alpha_i = \{s \in S : i \in \alpha(s)\}$. The **Partition Intersection Graph** of I is the vertex-colored graph $(G_I = (V, E), color)$ defined by $V = \{\alpha_i : \alpha \in C\}, E = \{(\alpha_i, \beta_j) : \alpha_i \cap \beta_j \neq \emptyset$, where $i \neq j$ if $\alpha = \beta\}$ and for $\alpha \neq \beta, color(\alpha_i) = color(\alpha_j) \neq color(\beta_s)$. Note that because the input I can have load greater than one, the coloring function *color* may not be proper. This definition of partition intersection graphs is an extension of the one in [7].

The main results leading to the algorithm can be paraphrased as follows:

- Let I be an input to the l-load perfect phylogeny problem. Then there is an l-load perfect phylogeny for I if and only if the partition intersection graph G_I has an l-triangulation.
- Given a graph G which is vertex-colored using k colors (not necessarily properly colored), we can determine in time polynomial in fixed k and l whether G has an l-triangulation and construct the l-triangulation when it does.

- Given an l-triangulation G' of G_I, we can construct an l-load perfect phylogeny in polynomial time.

As a consequence, we will provide an algorithm for determining if an l-load perfect phylogeny exists for k polymorphic characters defined on n species in $O((rk^3l^2)^{kl+1} + n(kl)^2)$ time.

5.2.2. *Characterization of l-triangulated graphs.* There is a well known characterization of triangulated graphs as intersection graphs of subtrees of a tree [4]. In this section, we will look at an extension of this particular characterization for l-triangulated graphs.

THEOREM 10. *Let $G = (V(G), E(G))$ be a vertex-colored graph. Then G is l-triangulated if and only if there exists a tree $T = (V(T), E(T))$ together with functions $\varphi : V(G) \to \{\text{subtrees of } T\}$ and $\phi : V(T) \overset{\text{bijection}}{\to} \{\text{maximal cliques of } G\}$ such that*

1. *$(v, w) \in E(G)$ iff $\varphi(v) \cap \varphi(w) \neq \emptyset$*
2. *$\varphi(v) = \{u \in V(T) : v \in \phi(u)\}$*
3. *$\forall v \in V(T), \phi(v)$ has at most l vertices of the same color*

THEOREM 11. *Given an instance I of the l-load perfect phylogeny problem, let G_I be the corresponding partition intersection graph. Then I has a solution iff G_I has an l-triangulation.*

5.2.3. *l-triangulating a vertex-colored graph.* In this section we turn to the problem of l-triangulating a vertex-colored graph. The solution to this problem makes use of several properties of triangulated graphs and also of a particular class of triangulated graphs called k-trees.

Further definitions: Triangulated graphs admit orderings, $v_1, v_2, \ldots, v_n$, on the vertex set such that for each i, $N_i = \Gamma(v_i) \cap \{v_{i+1}, v_{i+2}, \ldots, v_n\}$ is a clique [5]. These orderings are called **perfect elimination schemes**.

Consider a graph $G = (V, E)$ with $|V| = n \geq k$ that contains at least one k-clique. Such a graph G is a **k-tree** if the nodes of G can be ordered $v_1, v_2, \ldots, v_n$ whereby $\Gamma_G(v_i) \cap \{v_{i+1}, v_{i+2}, \ldots, v_n\}$ is a k-clique for all i with $1 \leq i \leq n - k$. A k-tree also has the following recursive definition: the complete graph on k vertices is a k-tree; if $G = (V, E)$ is a k-tree, and $S \subset V$ is a k-clique, then the graph formed by adding a new vertex v and attaching it to each vertex in S is also a k-tree. Each k-tree may be constructed using several different sequences of these operations. The initial set $S \subset V$ is called a **basis** for the k-tree.

A set S, of vertices, of a graph G is called a **vertex-separator** if $G - S$ is disconnected.

For a graph $G = (V, E)$ and vertex-separator $S \subset V$ with C a component of $G - S$, we define $\mathbf{C \cup cl(S)}$ to be the graph formed by adding to the subgraph of G induced by $C \cup S$ sufficient edges to make S into a clique. Let $G = (V, E)$ be a k-colored graph. We say that G is a **(k,l)-partition intersection graph** if (a) the maximum monochromatic clique size is l, and (b) G is edge covered by kl-cliques. Note that the maximum clique size in a (k, l)-partition intersection graph is kl.

The algorithm we present for l-triangulating a vertex-colored graph is based on dynamic programming. It is based on the algorithm of [8] for triangulating colored graphs. We will need the following lemma in our algorithm.

LEMMA 2. *Let G be a (k, l)-partition intersection graph. Then G can be l-triangulated if and only if there exists a set $K \subseteq V$ of size $(kl - 1)$ which is a*

separator for G such that for all components C of $G - K$, $C \cup cl(K)$ can be l-triangulated.

PROOF. It can be shown that G has an l-triangulation iff it has an l-triangulation G' which is a $(kl-1)$-tree. If such a G' exists, then G' has a separator of size $kl - 1$ which is a clique by [11]. The converse is straightforward. □

We are thus motivated to make the following definition:

DEFINITION 12. *Let $G = (V, E)$ be a vertex-colored graph with k colors and with maximum monochromatic clique size l. A potential basis for G', the l-triangulation of G, is a subset $V_0 \subseteq V$ such that (a) $|V_0| = kl - 1$ and (b) V_0 is a vertex separator for G. If $V_0 \subset V$ satisfies both these conditions then we say that V_0 is a* **potential basis** *for G, and call V_0 a* **pb**-*set.*

Our dynamic programming algorithm will solve the l-load problem when the input is a (k, l)-partition intersection graph. As our input graphs may not be (k, l)-partition intersection graphs, we need the following result:

LEMMA 3. *Let $G = (V, E)$ be vertex-colored with a coloring function color (using k colors) and assume that the maximum monochromatic clique size is l. Then there exists a (k, l)-partition intersection graph $G' = (V', E')$ such that the following is true:*

- *For every pb-set $S \subseteq V'$ containing $(k - 1)$ colors and every component C of $G' - S$, $C \cup S$ has all k colors present,*
- *G can be l-triangulated if and only if G' can be l-triangulated, and*
- *The number of vertices in G' is $|V| + |E|(kl - 2)$.*

We now have the basis for an algorithm for computing l-triangulations of vertex-colored graphs:

Algorithm B: l-triangulating k-colored graphs

Step 1:: Embed G in a (k, l)-partition intersection graph, G'.

Step 2:: Compute all *pb*-sets $V_0 \subseteq V(G')$, and all components C of $G' - V_0$. The subproblems $C \cup cl(V_0)$ are then bucket sorted by size.

Step 3:: Use dynamic programming to determine the answers for each subproblem in turn.

Step 4:: If there is a *pb*-set V_0 such that for all components C of $G' - V_0$, $C \cup cl(V_0)$ is has an l-triangulation, then return (Yes), else return (No).

It is clear that we need to indicate how we implement Step 3.

Solving Subproblems using Dynamic Programming: We have thus reduced the problem of determining whether the graph G can be l-triangulated to looking at graphs of the form $C \cup cl(S)$, where S is a *pb*-set, C is one of the components of $G' - S$, and we presume G' to be a (k, l)-partition intersection graph.

The following Theorem tells us when the l-triangulation can be done.

THEOREM 13. *Let $G = (V, E)$ be a (k, l)-partition intersection graph with $|V| \geq kl + 1$. Let S_0 be a pb-set and let C be a component of $G - S_0$. Then $C \cup cl(S_0)$ can be l-triangulated if and only if there exists some vertex v in C and a family of pb-sets F such that the following is true:*

1. *For each $M \in F$, $M \subset S_0 \cup \{v\}$, and M is a separator for $C \cup cl(S_0)$ and for G,*

2. *For each vertex* $x \in S_0$ *there is a* $M_x \in F$ *and a component* C_x *of* $G - M_x$ *and of* $C \cup cl(S_0) - M_x$ *such that* $|C_x| < |C|$ *and* $C_x \cup cl(M_x)$ *can be* l*-triangulated.*
3. *Every edge in* C *is in exactly one* C_x *given above.*

We can now conclude with our final theorem.

THEOREM 14. *Let* $G = (V, E)$ *be a* (k, l)*-partition intersection graph and let* $S \subset V$ *be a pb-set and* C *be a component of* $G - S$*. Then we can determine whether* $C \cup cl(S)$ *can be* l*-triangulated simply by knowing the "answer" for each smaller graph of the form* $C' \cup cl(S')$*, where* S' *is a pb-set and* C' *is a component of* $G - S'$*.*

We summarize with the following:

THEOREM 15. *Let* $G = (V, E)$ *be a* (k, l)*-partition intersection graph. We can in* $O(|V|^{kl+1})$ *time determine whether* G *can be* l*-triangulated, and produce the* l*-triangulation when it exists.*

5.2.4. *Summary of the algorithm to solve the* l*-load perfect phylogeny problem.* Given I, compute the Partition Intersection Graph, G_I, and embed G_I in a (k, l)-partition intersection graph G'_I. Use Algorithm B to determine if G'_I can be l-triangulated, and compute the triangulation G''_I if it exists. If there is no l-triangulation, Return No. Else, use G''_I to compute the l-load perfect phylogeny T.

THEOREM 16. *The* l*-load perfect phylogeny problem can be solved and the* l*-load perfect phylogeny constructed (when it exists) in* $O(nk^2l^2 + (rk^3l^2)^{kl+1})$ *time.*

Comment: In the case where individual load bounds l_c are given, the algorithm can be modified to run in $O(nL^2 + (rkL^2)^{L+1})$, where $L = \sum_{c \in C} l_c$.

5.3. Inferring Perfect Phylogenies from Mixed Data. In the previous section we presented two algorithms for inferring perfect phylogenies from polymorphic character data; these algorithms had running times which were exponential in L, where $L = \sum_{c \in C} l_c$, and l_c is the load bound for the character c. We can use these algorithms directly for sets of characters when some of the characters are monomorphic and some are polymorphic, but the expense would be too large. This follows since in typical data sets, the number of characters k is the largest parameter, often in the hundreds or thousands; since $L > k$, algorithms that are exponential in L are prohibitively costly. Instead, we propose a method which should be efficient when the number of monomorphic characters is sufficient to reduce the number of minimal perfect phylogenies to a small number. In practice, as the majority of the characters will be monomorphic, this is likely to be very efficient. The method we propose involves two steps, and is efficient when the number of minimal perfect phylogenies generated from the monomorphic characters is small.

Algorithm C:

Step 1: Infer all minimal perfect phylogenies from the monomorphic characters, using [**6**].

Step 2: Determine whether any of the minimal perfect phylogenies obtained in Step 1 can be refined so that each polymorphic character is convex on it within the specified load bound.

5.3.1. *Discussion of Step 1:* The algorithm in [**6**] has running time which is $O(2^{2r+r^2}k_m^{r+3} + Mk_mn)$, where M is the number of minimal perfect phylogenies and k_m is the number of monomorphic characters. This is theoretically expensive if r, the number of states, is too large; however, in practice, the algorithm works quickly as long as not too many of the characters have large number of states. Also, in practice, as long as the monomorphic characters are independent of each other and comprise a suitably large set, there will be very few perfect phylogenies. Thus, we expect Step 1 to be very fast, and to produce very few minimal perfect phylogenies.

5.3.2. *Discussion of Step 2:* We consider the following problem:

Problem: Refining a tree

Input:: Leaf-labelled tree T, and set C of polymorphic characters, each with an individual load bound.

Question:: Does a perfect phylogeny T' exist for the polymorphic characters, subject to the constraint that T' is a refinement of T?

Algorithm D:

For each internal $v \in T$ which has degree greater than 3, do:

1. Let $\Gamma(v) = \Gamma_1(v) \cup \Gamma_2(v)$ where $\Gamma_1(v)$ consists of all the neighbours of v which are leaves and $\Gamma_2(v)$ consists of all the non-leaf neighbours of v. For each $u_j \in \Gamma_2(v)$ add a new node w_j on the edge (v, u_j). Compute the labelling of w_j so as to make every character convex (each character must contain every state that appears on both sides of w_j).
2. If some new node has a load for a character that exceeds the stated bound for that character, RETURN(No). Let $S_v = \Gamma_1(v) \cup \{w_j | w_j$ is a new node and w_j is a neighbor of $v\}$. Use any of the algorithms from Section 5 to determine if there is a perfect phylogeny for (S_v, C). If any (S_v, C) fails to have a perfect phylogeny then RETURN(No), else RETURN(yes).

THEOREM 17. *Algorithm D correctly determines whether a perfect phylogeny T' exists refining T within the stated load bounds, and can be modified to produce the perfect phylogeny T' in time $min\{O(r^{L+1}Ln^2), O(n^2L^2 + n(rkL^2)^{L+1})\}$.*

In *Algorithm D*, if $|S_v|$ is small then it may be cheaper in practice to look at all possible leaf-labelled topologies on S_v rather than use the algorithms of Section 5 to determine the existence of perfect phylogenies on S_v.

6. Polymorphism in Linguistics

Properly chosen and encoded characters in Linguistics have been shown to be convex on the true tree, so that with proper scholarship we should be able to infer a *perfect phylogeny.* In recent work on an Indo-European data set, [**13**] found that there was extensive presence of polymorphic characters (35 out of 220 characters were polymorphic). The degree of polymorphism for each polymorphic character could be determined from the data with high confidence, so that the question of inferring the correct tree amounted to determining if a perfect phylogeny existed in which each character was permitted a maximum degree of polymorphism (i.e. load) on the tree. Figure 1 shows the tree that they now posit. This was obtained using Algorithm D, which we described earlier. This tree is infact different from their earlier hypothesis (which was presented at the NAS Symposium on the Frontiers of Science in November 1995) and also the tree that was presented in [**3**]. This

tree has been obtained as a result of using more data. This tree shows a limited support for Indo-Hittite hypothesis (that is, the first subfamily to break off from the root of the Indo-European evolutionary tree should be the Anatolian branch), moderate support for the Italo-Celtic hypothesis (that is, Italic and Celtic should be sisters within the tree, and without a third sister), and a significant support for a subgroup of Greek and Armenian.

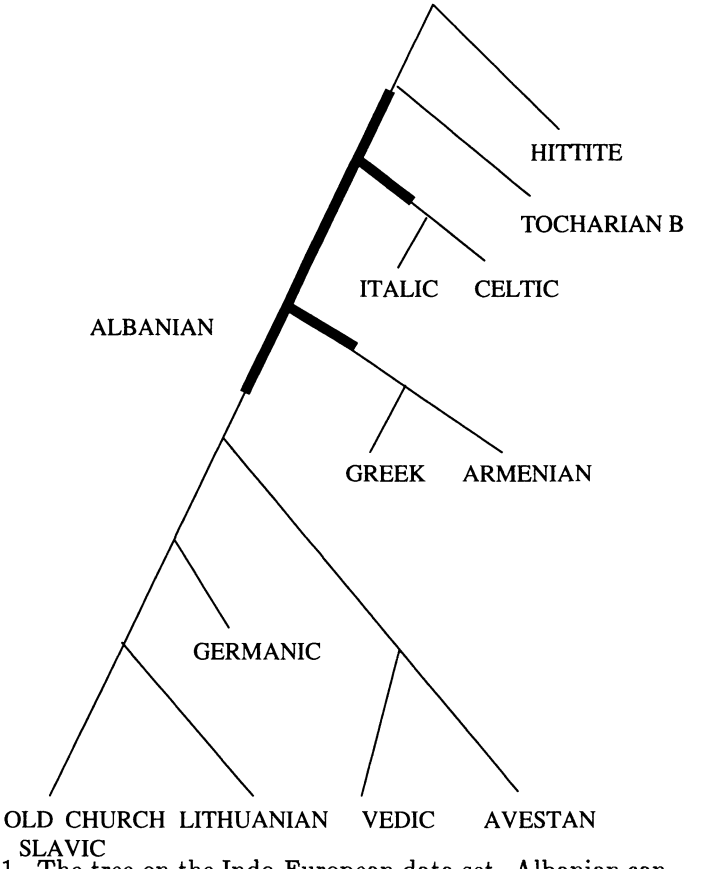


FIGURE 1. The tree on the Indo-European data set. Albanian can be on any of the thick edges. The tree just indicates a rooted topology without any edge lengths.

7. Conclusion

In this paper we introduced an algorithmic study of the problem of inferring evolutionary trees in the presence of polymorphic characters. The results of our analysis of the data set studied by Warnow, Ringe and Taylor has led to a new hypothesis for the evolution of the Indo-European languages.

References

[1] R. AGARWALA AND D. FERNÁNDEZ-BACA, *Fast and Simple Algorithms for Perfect Phylogeny and Triangulating Colored Graphs.* To appear in the special issue on *Algorithmic Aspects of Computational Biology* of International Journal of Foundations of Computer Science. Available as DIMACS technical report TR94-51.

[2] H. BODLAENDER, M. FELLOWS, AND T. WARNOW, *Two strikes against perfect phylogeny*, In Proceedings of the 19th International Colloquium on Automata, Languages, and Programming, Springer Verlag, Lecture Notes in Computer Science (1992), pp. 273–283.

[3] M. BONET, C. PHILLIPS, T. WARNOW AND SHIBU YOOSEPH, *Constructing evolutionary trees in the presence of polymorphic characters*, In Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing, pages 220-229, Philadelphia, Pennsylvania, 22-24 May 1996.

[4] P. BUNEMAN, *A characterization of rigid circuit graphs*, Discrete Math 9 (1974), pp. 205-212.

[5] M. C. GOLUMBIC, *Algorithmic Graph Theory and Perfect Graphs*, 1980 Academic Press Inc.

[6] S. KANNAN AND T. WARNOW, *A fast algorithm for finding and enumerating perfect phylogenies*, Proc. 6th Annual ACM/SIAM Symposium on Discrete Algorithms, 1995, San Francisco.

[7] F. R. MCMORRIS AND C. A. MEACHAM, *Partition Intersection Graphs*, Ars Combinatorica, 16(1983), pp. 135-138.

[8] F. R. MCMORRIS, T. WARNOW, AND T. WIMER, *Triangulating vertex colored graphs*, SIAM J. on Discrete Mathematics, Vol. 7. No. 2, (1994), pp. 296-306.

[9] M. NEI, *Molecular Evolutionary Genetics*, Columbia University Press, New York. 1987.

[10] D. RINGE, personal communication, 1995.

[11] D.J. ROSE, *On simple characterization of k-trees*, Discrete Math., 7 (1974), pp. 317-322.

[12] M. A. STEEL, *The complexity of reconstructing trees from qualitative characters and subtrees*, Journal of Classification, 9 (1992), pp. 91–116.

[13] T. WARNOW, D. RINGE AND A. TAYLOR, *A character based method for reconstructing evolutionary history for natural languages*, Tech Report, Institute for Research in Cognitive Science, 1995, and *Proceedings 1996 ACM/SIAM Symposium on Discrete Algorithms.*

DEPARTMENT OF COMPUTER SCIENCE, UNIVERSIDAD POLITECNICA DE CATALUÑA, SPAIN.
E-mail address: bonet@goliat.upc.es

SANDIA NATIONAL LABS, ALBUQUERQUE, NM, USA.
E-mail address: caphill@cs.sandia.gov

DEPARTMENT OF COMPUTER AND INFORMATION SCIENCE, UNIVERSITY OF PENNSYLVANIA, PHILADELPHIA, PA, USA.
E-mail address: tandy@central.cis.upenn.edu

DEPARTMENT OF COMPUTER AND INFORMATION SCIENCE, UNIVERSITY OF PENNSYLVANIA, PHILADELPHIA, PA, USA.
E-mail address: yooseph@gradient.cis.upenn.edu

This page intentionally left blank

https://doi.org/10.1090/dimacs/037/04

DIMACS Series in Discrete Mathematics
and Theoretical Computer Science
Volume **37**, 1997

Reconciled Trees and Incongruent Gene and Species Trees

Roderic D. M. Page and Michael A. Charleston

ABSTRACT. We present a method for visualising and quantifying the relationship between a pair of gene and species trees that constructs a third tree termed the reconciled tree. Given a gene tree and a species tree the reconciled tree represents the history of the gene tree embedded within the species tree. The reconciled tree is constructed from one or more subtrees of the species tree, and contains the gene tree as a subtree. The relationship between the gene and species tree can be expressed in terms of the number of gene duplications and gene losses required to construct the reconciled tree. This number can be used as an optimality criterion for selecting the species tree that best accounts for the observed gene tree.

Introduction

The key assumption that motivates molecular systematics is that evolutionary trees for genes also contain information about the evolutionary relationships of organisms. Indeed, it is often assumed that gene trees and species trees are isomorphic; once the gene tree is obtained the species tree can be obtained simply by relabelling the leaves of the gene tree with the names of the corresponding species. However, two observations contradict this assumption: (1) species may contain more than one copy of the same gene, and (2) different gene trees may imply different species trees. If two or more copies of the gene are sequenced then relabelling the gene tree with the species names will result in some species occurring more than once. In this case there is no longer a one-to-one correspondence between the gene and species trees, raising the problem of how to extract the latter from the former. If different gene trees support different species trees (that is, the gene trees are incongruent) then this raises the question of how to choose among these alternative species trees.

Our goal in this paper is to outline an approach for visualising the relationship between gene and species trees. This method employs a third tree which we call the reconciled tree. The term comes from Goodman's goal [7] of reconciling incongruent gene and species trees. The reconciled tree corresponds to a map between the gene and species tree which associates each node in the gene tree with a node in the species tree. Unless the reconciled tree is identical to, or a subtree of, the species

1991 *Mathematics Subject Classification.* Primary 92B10; Secondary 92D15.

Key words and phrases. gene tree, heuristic search, phylogeny, reconciled tree.

This work was supported by NERC grant GR3/1A095 to the first author. RDMP thanks Boris Mirkin, Ilya Muchnik and Oliver Eulenstein for discussions while he was a visitor at DIMACS.

tree then the reconciled tree has associated with it a cost that is the sum of the number of gene duplication and gene loss events required to reconcile the gene and species trees. Given that we can compute the "cost" of reconciling a gene and species tree, this cost can be used as an optimality criterion for choosing the species tree that yields the least costly reconciled tree for a given gene tree. Because of the vast number of evolutionary trees for even a few species [**6**] we will typically need to rely on heuristics to search for optimal species trees. We outline the use of techniques for characteristing the search landscape that allow insight into the performance of various search strategies for finding the optimal species tree. We use this technique to reanalyse the 53 gene trees studied by Guigó *et al.* [**8**].

To avoid potential confusion it is useful to clarify how this approach differs from consensus methods [**1, 11**] which it superficially resembles. Consensus methods operate on two or more trees with the same terminal labels and are used to display the extent to which two or more trees agree on relationships among the same set of objects. Reconciled trees, however, operate on trees for different entities (e.g., genes and organisms) which are in some sense associated (it is this association that allows us to compare the trees by establishing a relationship between the terminal labels in the two trees). Furthermore, a reconciled tree results from embedding one tree into another. In an important sense, which we elaborate on below, the reconciled tree combines information from both the trees being compared, unlike consensus methods which represent only shared information.

Maps between trees

Central to the concept of a reconciled tree is the notion of a map between two trees. This idea was first introduced by Goodman *et al.* [**7**] and has recently attracted renewed attention [**4, 8, 13, 16**]. For simplicity, let us initially assume that we have only a single gene in each of our study species. To distinguish between genes and species we will use the convention of labelling species by the letters $a, b, c, \ldots$, and the genes from those species by $1, 2, 3, \ldots$, where gene 1 is from species a, gene 2 from species b, and so on. Let G be a binary gene tree for n sequences obtained from n species, and S be the binary species tree (Figure 1).

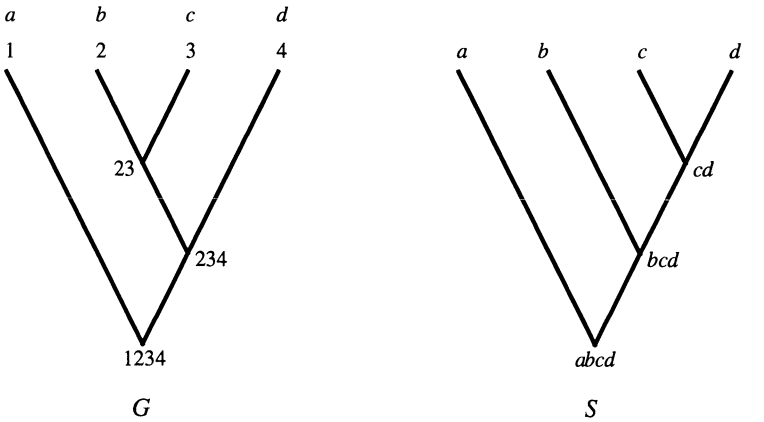


FIGURE 1. Example gene and species trees G and S

For any node $g \in G$, let $\eta(g)$ be the set of species in which occur the extant genes descendant from g. Also, for any $g \in G$, let $M(g) \in S$ be the smallest node in S that includes g, that is the smallest cluster satisfying $\eta(g) \subseteq M(g)$. The node $M(g) \in S$ corresponds to the most recent common ancestor of all the species in which either g (if g is a leaf) or all the genes descendant from g occur. The map between the internal nodes in the two trees in Figure 1 is shown in Figure 2.

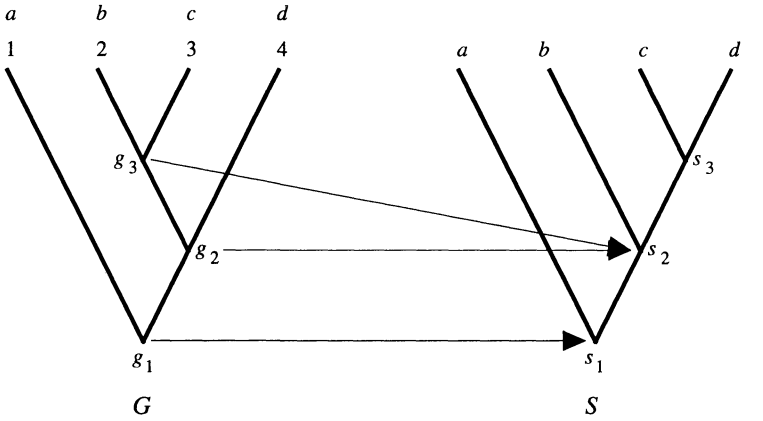


FIGURE 2. Mapping from gene tree G into species tree S.

Constructing a map amounts to finding for each node $g \in G$ the most recent common ancestor of all the species containing genes that descend from g. In the example in Figure 2, the descendants of g_3 are found in species b and c, hence g_3 corresponds to the most recent common ancestor of these species in S, namely s_2.

Duplications. If each g has a unique image $M(g)$ then G and S are said to be *consistent.* If G and S are not consistent then there will be cases where more than one node in G maps onto the same node in S. These cases are termed duplications.

DEFINITION 1. *A* duplication *is an internal node $g \in G$ for which $M(g) = M(g_l)$ or $M(g) = M(g_r)$, where g_l and g_r and the left and right children of g, respectively.*

For the trees in Figure 2, $M(g_2) = M(g_3) = s_2$, hence there is a duplication at g_2.

Reconciled trees

Reconciling two incongruent gene and species trees requires postulating a combination of gene duplications and losses [**7**]. A duplication results in two copies of the gene, hence we would expect all the descendants of the species lineage in which the duplication took place to possess those two copies. If they do not then we must postulate gene losses. Figure 3 shows the gene tree G from Figure 1 embedded in its species tree S, and the corresponding reconciled tree.

The duplication at g_2 results in two pairs of gene lineages. Three gene losses (one each in species b, c, and d) are required to account for the absence of one or other of the two gene lineages in those species. If there had been no gene losses then the gene tree would comprise seven leaves. This complete gene tree is the reconciled tree [**14, 16**].

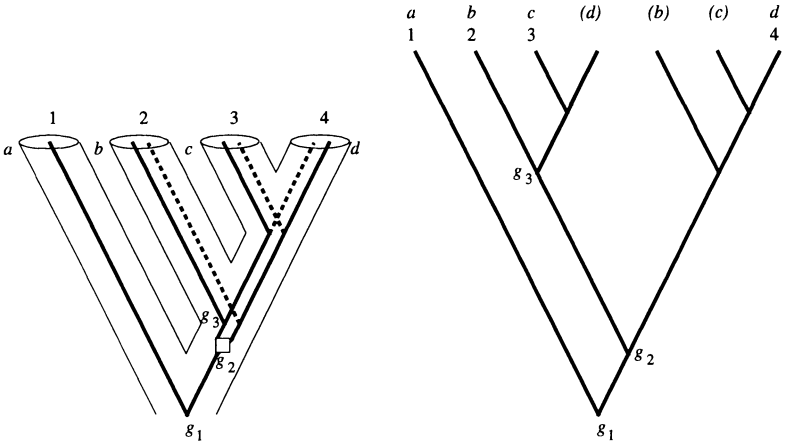


FIGURE 3. Embedding of a gene tree into a species tree, and the "unfolded" gene tree forming the reconciled tree.

The reconciled tree R has two important properties which allow it to depict the relationship between the gene and species tree. The first property is that the observed gene tree is a subtree of the reconciled tree (Figure 3). The second property is that if we label each leaf of the reconciled tree with the corresponding species label then the clusters of the reconciled tree are all clusters of the species tree (Figure 4).

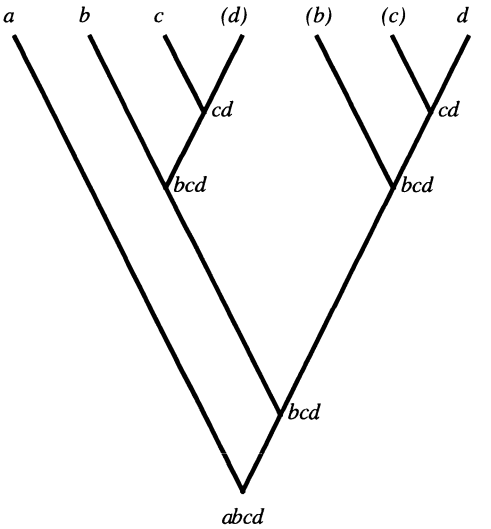


FIGURE 4. Relabelled reconciled tree.
Each cluster in the reconciled tree is also a cluster in the species tree.

However, whereas in S the clusters of any two children of a node $s \in S$ are disjoint, in the reconciled tree any two child clusters of an internal node $r \in R$ are either disjoint or identical. The later case corresponds to a duplication.

Computing numbers of losses. If the gene tree is a strict subtree of the reconciled tree, then there must have been gene losses. To compute the number of losses we can color the leaves of the reconciled tree with either $\{1\}$ (presence) or $\emptyset$ (absence) of the gene. Each internal node g is assigned the color $c_g = c_{g_l} \cup c_{g_r}$. If $c_{g_l} \cap c_{g_r} = \emptyset$ then one of the node's children has lost a gene. Hence we can compute the number of losses in a single post-order traversal (i.e., from leaf vertices to root) of the reconciled tree. In examples we have tried this procedure finds the same number of losses as the formulae in [**8**] and [**13**] which compute losses as a function of the number of nodes between $M(g)$ and $M(g_l)$ and $M(g_r)$ for each internal node in the gene tree. For formal proof of the equivalence of these measures see Eulenstein, Mirkin and Vingron (in this volume).

Constructing the reconciled tree. In a reconciled tree a duplication is indicated by a node whose two children have the same cluster (Figure 4). Hence the reconciled tree is assembled from subtrees of S. The following is a sketch of the algorithm in [**16**] for constructing a reconciled tree R for gene tree G and species tree S:

: Step 1. Let $R = S$. Color each leaf in R with $\{1\}$.

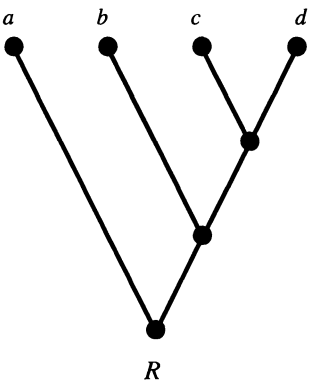


FIGURE 5. Step 1: Each leaf is colored with $\{1\}$.

: Step 2. Traverse G in preorder (i.e., from root to leaf vertices). For each node g that is a duplication go to step 3. If the tree has been completely traversed go to step 5.

: Step 3. Find the node $r \in R$ that corresponds to $M(g)$. Copy the subtree in S rooted at $M(g)$ and add this to R below r. Inserting this subtree creates an additional node x which corresponds to the gene duplication at g. Figure 6 shows this step for the trees in Figure 1.

: Step 4. Color the descendants of the two subtrees rooted at x to reflect the presence of the gene. Let x_l and x_r be the left and right children of x, respectively. For each terminal descendant i of $x_l \in R$, if $i \in \eta(g_l)$ then color it $\{1\}$; otherwise color it $\emptyset$. We reverse the colors for x_r, the right child of x.

The internal nodes are colored using the rule $c_g = c_{g_l} \cup c_{g_r}$. The result of this step for the trees in Figure 1 is shown in Figure 7, where $\bullet = \{1\}$ and $\circ = \emptyset$. Return to step 2.

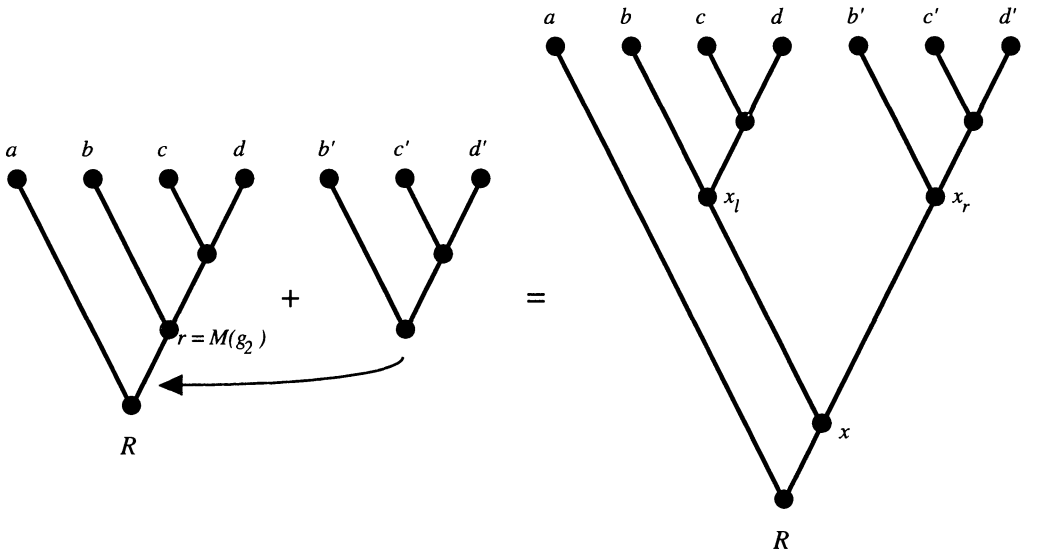


FIGURE 6. Step 3: Adding the subtree rooted at $M(g_2)$.

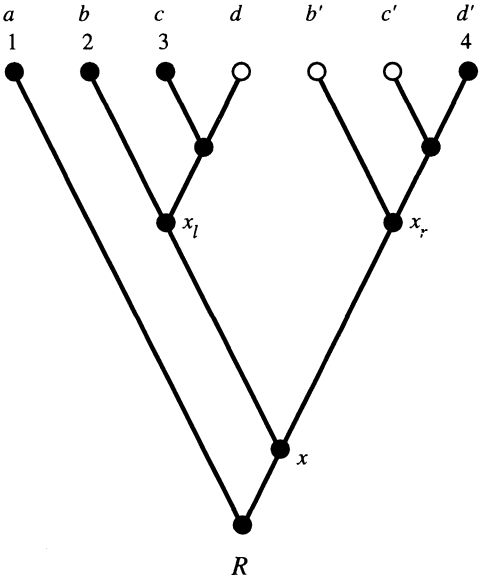


FIGURE 7. Step 4: Coloring the reconciled tree.
The reconciled tree with nodes colored to indicate the presence (•) or absence (○) of a gene lineage.

: Step 5. Compute the number of losses as described above.

Properties of reconciled trees. The formal properties of reconciled trees have been little explored. One possibility for their exploration is modelling them using multisets [**12**] as sketched by Page [**16**]. Clearly the reconciled tree for a given G and S is unique, but this is not sufficient to guarantee the following conjectures:

1. The reconciled tree $R(G, S)$ minimises the sum d of duplications.
2. The reconciled tree $R(G, S)$ minimises the sum $(d + l)$ of duplications and losses.
3. For a given G, all S that minimise $(d + l)$ also minimise d.

These questions are believed to be open at the time of writing, and invite investigation.

Note that the mathematical results presented here are not restricted to binary gene and species trees. The mapping and construction of reconciled trees is well-defined and consistent for polytomous trees. (A polytomous tree is one which contains internal vertices with degree greater than three.) However, interpretation of such trees must be cautious. There are two possible interpretations of polytomies: *hard* and *soft* [**10**].

All the immediate descendents of a hard polytomy are assumed to have diverged at the same time, whereas in a soft polytomy the descendants are assumed to have diverged at different, unknown, times.

Embedding subtrees. If we are comparing more than one gene tree with a species tree then it will often be the case that not all the genes are known in all the species of interest. While the algorithms for mapping two trees and for constructing the reconciled tree are still applicable in this case, the number of losses computed needs to be interpreted carefully. For example, in the algorithm given above a gene tree for four genes may be perfectly consistent with a larger species tree (on $n > 4$ species say), but the lack of genes in the remaining $(n - 4)$ of those species will be counted as losses. Given the uneven taxonomic sampling in the sequence data bases (e.g., the predominance of mammals among the 101 genes in the SWISPROT data base listed by Guigó *et al.* [**8**]) a more reasonable interpretation may be that these species simply have not been sequenced for that gene.

One solution to this problem is to construct the reconciled tree from the subtree that results from pruning the species for which the gene locus is unknown. An alternative is to introduce a third color, "?", for those leaves in the reconciled tree that correspond to species that lack any representatives of the gene, that is $s \in (S \setminus G)$. If neither child of g has color "?" then the rule presented in section 3.1 still applies. However, if either one of the other child, but not both, is "?" then g takes the color of the other child. If both children are "?" then $g =$ "?" The advantage of constructing the reconciled tree for the complete set of species is that the reconciled tree again highlights those species which we might expect to harbour undiscovered sequences related to the subset of known sequences. For a different treatment of the same problem see Mirkin et al [**13**].

Kinds of duplications. For ease of presentation so far we have considered only the case where each species has a single gene, which is the only case considered by Guigó *et al.* [**8**] and Mirkin et al [**13**]. However, we may have gene trees in which more than one sequence is available from the same species. In this instance there will be one or more $g \in G$ where $\eta(g_l) \cap \eta(g_r) \neq \emptyset$. However, we can show that for any such node g, either one or both of its children will map onto the same node in the species tree and hence be correctly interpreted as a gene duplication:

LEMMA 1. *For $g \in V(G)$ ($V(G)$ is the vertex set of G) with children g_l and g_r such that $\eta(g_l) \cap \eta(g_r) \neq \emptyset$, the embedding mapping M described above is such that either $M(g) = M(g_l)$ or $M(g) = M(g_r)$ or both.*

PROOF. Let $M(g_l) = x$, $M(g_r) = y$, and $M(g) = z$. Then x is the smallest superset in S of $\eta(g_l)$, y is the smallest superset in S of $\eta(g_r)$, and z the smallest superset in S of $\eta(g)$. Choose $a \in \eta(g_l) \cap \eta(g_r)$. Suppose that $z \notin \{x, y\}$. Then we must have $z \supset x$ and $z \supset y$ since S is a tree. Since $a \in g_l$ then $x \cap y = \emptyset$, a contradiction. □

Heuristic searches for optimal species trees

The cost of mapping a gene tree into a species tree can be used as an optimality criterion for choosing among alternative species trees. If the species tree is unknown then a natural candidate for it is the tree that yeilds the least costly reconciled tree [**8, 16, 19**]. Given the large numbers of possible trees for even moderate numbers of species [**6**] we will usually have to rely on heuristics which do not guarantee to find the globally optimal solution.

One approach is to search tree space using tree perturbations such as the well-known Nearest Neighbour Interchange (NNI) [**23**]. An initial starting species tree is chosen and its cost is computed by reconciling it with the gene tree. The start tree is then perturbed in search of a better tree. If one is found, the search continues from the better tree, repeating until no perturbation produces an improvement. This strategy of hill climbing is sensitive to the initial starting tree, and to the conformation of the landscape for the problem instance [**3**]. In particular, if the search landscape has several locally optimal peaks the heuristic search may find a species tree which is locally optimal but far removed from the global optimum. This problem can be clearly illustrated using the recent study by Guigó *et al.* of 53 genes from a range of eukaryotes [**8**].

Eukaryote example. Guigó *et al.* took 53 gene trees and searched for the least-cost species tree with which to reconcile them.

Tree perturbations and the search landscape. Taking a landscape approach to the investigation of tree space is a fruitful method of determining the nature of phylogenetic signal in a data set. In this section we describe how landscape-based methods were used to assess the ruggedness of the landscape of the solution space to this problem instance.

We performed 50 simple hill-climbing heuristic searches, with randomly chosen starting trees, using each of two sets of tree perturbations to move between estimated species trees. At each step in a search, the current tree would be perturbed until either a better tree (with lower total cost) was found, or all instances of the perturbation were tried without success, at which point the search would be halted. The initial trees, chosen at random, were identical for the two search strategies.

The first tree perturbation used was Nearest Neighbour Interchange (NNI, [**23**]) by itself; the second set of perturbations (ALT) alternated between NNI and Cut and Paste (C&P, also known as Subtree Pruning and Regrafting or SPR [**22**]). Thus the adjacencies in the second landscape included those in the first.

The NNI search was found to be markedly poorer in recovering least-cost solutions (trees) to this problem instance: in all 50 NNI runs the best tree found, of cost 171, was obtained just once. The best tree found using the ALT method had a cost of 159 (36 duplications and 123 losses), which was obtained 6 times. Note also that all but one of the NNI searches was less successful than all the ALT searches.

Figure 8 shows the costs of the best trees found in each of the searches, for NNI and ALT methods, plotted against maximal steepest climb length [**3**]. From this figure we can deduce that the landscape induced by the NNI adjacencies is more "rugged" than that induced by (NNI + C&P) as with the ALT search.

Since our best tree found was still obtained only 6 times in the 50 runs, we must stress the importance of performing multiple searches from random starting points [**9**]. We cannot begin to have confidence in the global optimality of our best solutions found, until we have encountered them many times.

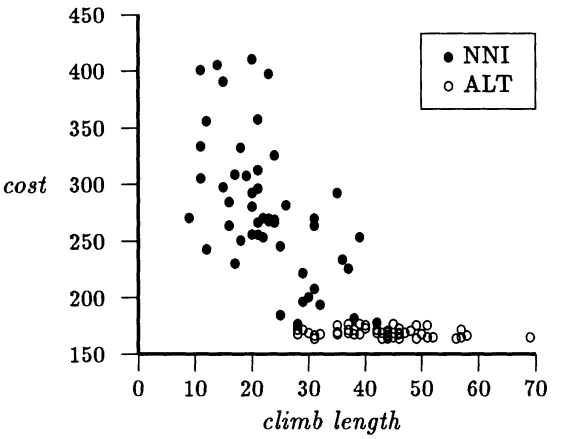


FIGURE 8. The maximal steepest climb lengths for 50 NNI and 50 ALT searches.

In the above figure we see the markedly poorer performance of simple NNI searching compared with the ALT search (q.v.). In 50 NNI searches none found the best found with the ALT search.

Guigó *et al.*'s preferred species tree is shown in Figure 9(a), for which they found 46 duplications and 101 losses. Using the algorithm for constructing reconciled trees we also found 46 duplications but an additional 44 losses, for a total cost of $46+145 = 191$. The same cost was found using formula from [**13**], hence we suspect that Guigó *et al.*'s value for the losses is an error. Guigó *et al* also report that their best species tree is wholly consistent with 18 of the gene trees, however we find it is consistent with only 17.

Our best tree found using the ALT searches has a cost of 159 (36 duplications + 123 losses) and is consistent with 25 of the gene trees (Figure 9(b)).

Using Page's program COMPONENT [**15**] we found that the Guigó *et al.* tree is 7 NNI steps from their seed tree (Figure 9(c)). This distance is substantially shorter than most of our NNI climbs, but comparison of the searches on this basis is confounded by the difference in counting methods of the authors and of Guigó *et al.*. It is interesting to note that the best tree found (Figure 9(b)) in the ALT searches is 13 NNI steps from their seed tree, and 15 NNI steps from their best tree found, so in a very loose sense the heuristic search strategy adopted by Guigó *et al* went in the wrong "direction" from the seed tree.

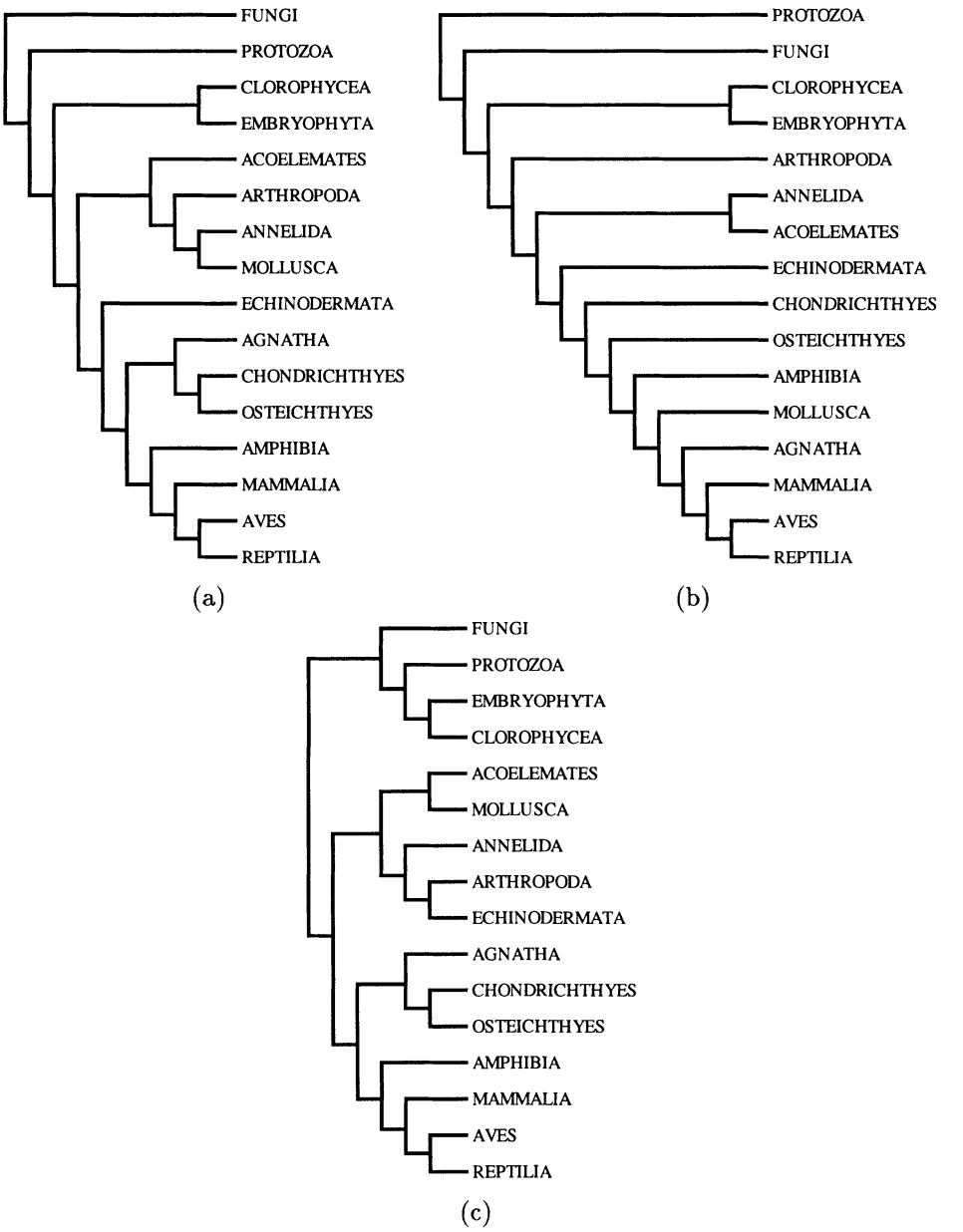


FIGURE 9. Three species trees for the 53 gene trees: (a) Guigó *et al.*'s tree; (b) our tree; (c) Guigó *et al.*'s seed tree.

The best trees found. A more thorough search led to the discovery of 11 more trees, each requiring 36 duplications and 123 losses, and which differed only slightly from each other. The most biologically reasonable one is that shown in Figure 9(b). The Adams consensus tree [**2**] of these 12 trees is shown in Figure 10(a), and the strict consensus is in Figure 10(b).

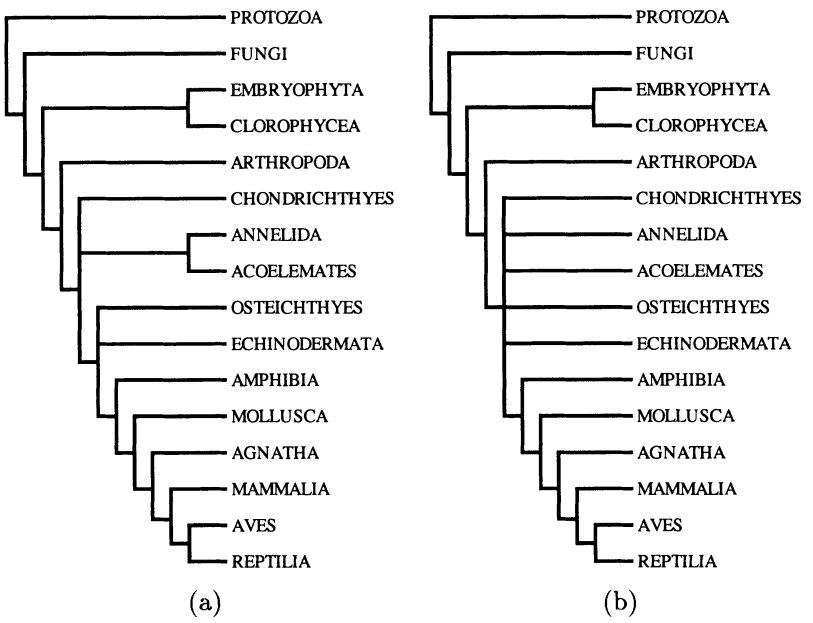


FIGURE 10. Adams and Strict consensus trees of the 12 least-cost species trees found from 53 gene trees.

Note that of the original 53 gene trees, only 4 had a mollusc sequence, and only one had an Agnathan, so it is perhaps not surprising that it is these two taxa whose position is so odd in the optimal trees we found.

Limitations

Allele phylogenies and coalescence. In this paper we have focussed on interpreting reconciled trees in terms of gene duplications and losses. Descendants of one or more gene duplications are paralogous [5]. However, orthologous sequences (which by definition have not undergone gene duplication) may also be present in multiple copies (alleles) and may yield gene trees which are discordant with the species tree. In this context "duplications" inferred by the reconciled tree are not literally gene duplications; rather they represent coalescences (instances of common ancestry) of intraspecific allele lineages. Rather than numbers of duplications and lossess it may be more biologically meaningful to count other aspects of the reconciled tree, such as the number of times a pair of alleles from two different species fail to coalesce in the immediate ancestor of those species. Failures of the alleles to coalesce, depth to coalescence, and numbers of gene lineages present on each edge in the species tree are among the parameters that could be readily measured.

Horizontal Transfer. From a biological perspective perhaps the greatest limitation of this approach is that it excludes any possibility of horizontal transfer of genes between different species lineages [21]. Reconciled trees require that a species always acquires its genes from its immediate ancestors, whereas horizontal transfer implies that a species may have acquired a gene from another, contemporaneous

lineage. Horizontal transfer introduces new complications because we are no longer simply interested in embedding one tree inside another. In particular, horizontal transfer establishes links between edges of the species tree. Given that horizontal transfer must take place between contemporaneous lineages not all the possible pairs of edges will be valid horizontal transfers. Page [**17**] pointed out that transfers cannot take place between a lineage and either its descendants or its ancestors. However, by itself this rule is inadequate to ensure that only logically valid horizontal transfers are postulated [**20**]. Consider the example shown in Figure 11 of a species tree with two possible horizontal transfers indicated.

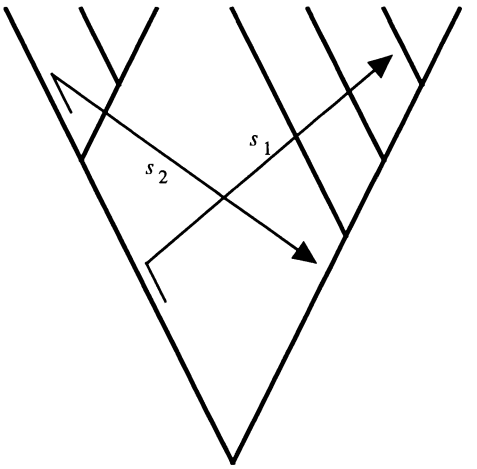


FIGURE 11. A pair of incompatible horizontal transfer events

In each case the transfer is between a pair of edges where neither edge is ancestral to the other, satisfying the rule in [**17**]. However, considered together these two horizontal transfers are mutually incompatible, as they stand. There is no ordering of the internal nodes of the species tree that will allow both switches to take place without one transfer going forward or back in time (Figure 12).

Horizontal transfer introduces additional complexity because we have to consider the relative (temporal) order of internal nodes in the evolutionary tree. The challenge is to develop methodologies which are capable of dealing with all the complexities introduced by horizontal transfers. This is being undertaken by the authors and is intended to be included in the next release of TREEMAP [**18**].

Conclusions

Reconciled trees are a simple way to visualise the relationship between a gene and a species tree. By displaying the complete history of the gene they allow us to see where gene duplications (both directly observed and inferred) occurred, and which species might yield further sequences of the same gene family. For these reasons we find them more intuitive than the labelling scheme adopted by Mirkin *et al.* [**13**]. The reconciled tree suggests a straightforward measure of the degree of fit between a gene tree and a species tree, namely the number of gene duplications and gene losses required to reconcile the two trees. This measure can be used as an optimality criterion for selecting among competing species trees. However, searches using this criterion must be conducted with care in order to avoid suboptimal species trees.

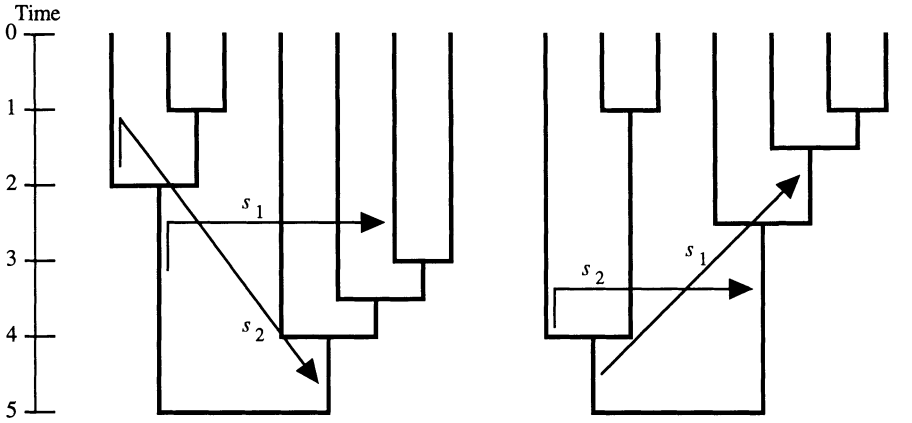


FIGURE 12. Two temporal orderings of the internal nodes of the tree in Figure 11

The horizontal transfers s_1 and s_2 are incompatible because there is no ordering of the internal nodes of the tree in Figure 11 that will ensure that both s_1 and s_2 take place between contemporaneous taxa.

References

[1] E. N. Adams. Consensus techniques and the comparison of taxonomic trees. *Systematic Zoology*, 21:390–7, 1972.

[2] E. N. Adams. N-trees as nestings: Complexity, similarity, and consensus. *Journal of Classification*, 3:299–317, 1986.

[3] M. A. Charleston. Towards a characterization of landscapes of combinatorial optimisation problems, with special reference to the phylogeny problem. *Journal of Computational Biology*, 2:439–50, 1995.

[4] O. Eulenstein and M. Vingron. On the equivalence of two tree mapping measures. *Arbeitspapiere der GMD, Germany*, 986, 1995.

[5] W. M. Fitch. Distinguishing homologous from analogous proteins. *Systematic Zoology*, 19:99–113, 1970.

[6] L. R. Foulds and R. W. Robinson. Enumeration of binary phylogenetic trees. In *Combinatorial Mathematics IX*, Lecture Notes in Mathematics 884, pages 187–202. Springer, Berlin, 1980.

[7] M. Goodman, J. Czelusniak, G. W. Moore, A. E. Romero-Herrera, and G. Matsuda. Fitting the gene lineage into its species lineage: A parsimony strategy illustrated by cladograms constructed from globin sequences. *Systematic Zoology*, 28:132–68, 1979.

[8] R. Guigó, I. Muchnik, and T. F. Smith. Reconstruction of ancient molecular phylogeny. *Molecular Phylogenetics and Evolution*, 6:189–213, 1996.

[9] D. R. Maddison. The discovery and importance of multiple islands of most parsimonious trees. *Systematic Zoology*, 40:315–28, 1991.

[10] W. Maddison. Reconstructing character evolution on polytomous cladograms. *Cladistics*, 5:365–377, 1989.

[11] T. Margush and F. R. McMorris. Consensus n-trees. *Bulletin of Mathematical Biology*, 43:239–44, 1981.

[12] N. Minaka. Cladograms and reticulated graphs: A proposal for graphic representation of cladistic structures. *Bulletin of the Biogeographic Society of Japan*, 45:1–10, 1990.

[13] B. Mirkin, I. Muchnik, and T.F. Smith. A biologically consistent model for comparing molecular phylogenies. *Journal of Computational Biology*, 2:493–507, 1995.

[14] R. D. M. Page. Component analysis: A valiant failure? *Cladistics*, 6:119–36, 1990.

[15] R. D. M. Page. COMPONENT. Version 2.0, 1993. Tree comparison software for Microsoft®/Windows™.

[16] R. D. M. Page. Maps between trees and cladistic analysis of historical associations among genes, organisms, and areas. *Systematic Biology*, 43:58–77, 1994.

[17] R. D. M. Page. Parallel phylogenies: Reconstructing the history of host-parasite assemblages. *Cladistics*, 10:155–73, 1994.

[18] R. D. M. Page and M. A. Charleston. TREEMAP program. Version 2.0. Platforms: Microsoft®/Windows™, Macintosh™. In prep.

[19] R. D. M. Page and M. A. Charleston. From gene to organismal phylogeny: Reconciled trees and the gene tree/species tree problem. *Molecular Phylogenetics and Evolution*, (in press).

[20] F. Ronquist. Reconstructing the history of host-parasite associations using generalised parsimony. *Cladistics*, 11:73–89, 1995.

[21] M. W. Smith, D.-F. Feng, and R. F. Doolittle. Evolution by acquistion: The case for horizontal gene transfers. *Trends in Biochemical Sciences*, 17:489–93, 1992.

[22] D. L. Swofford and G. J. Olsen. Phylogeny reconstruction. In David M. Hillis and C. Moritz, editors, *Molecular Systematics*, chapter 11, pages 411–501. Sinauer Associates, Sunderland, 1990.

[23] M. S. Waterman and T. F. Smith. On the similarity of dendrograms. *Journal of Theoretical Biology*, 73:789–800, 1978.

DIVISION OF ENVIRONMENTAL AND EVOLUTIONARY BIOLOGY, INSTITUTE OF BIOMEDICAL AND LIFE SCIENCES, UNIVERSITY OF GLASGOW, GLASGOW G12 8QQ, SCOTLAND, UNITED KINGDOM., PHONE: 44-141-330-4778, FAX: 44-141-330-5971

E-mail address: r.page@bio.gla.ac.uk

DIVISION OF ENVIRONMENTAL AND EVOLUTIONARY BIOLOGY, INSTITUTE OF BIOMEDICAL AND LIFE SCIENCES, UNIVERSITY OF GLASGOW, GLASGOW G12 8QQ, SCOTLAND, UNITED KINGDOM., PHONE: 44-141-330-5346, FAX: 44-141-330-5971

E-mail address: m.a.charleston@bio.gla.ac.uk

https://doi.org/10.1090/dimacs/037/05

DIMACS Series in Discrete Mathematics
and Theoretical Computer Science
Volume **37**, 1997

Comparison of Annotating Duplication, Tree Mapping, and Copying as Methods to Compare Gene Trees with Species Trees

Oliver Eulenstein, Boris Mirkin, and Martin Vingron

ABSTRACT. This paper reviews the authors' work on the idea of employing the mechanism of gene duplication in explaining the differences between a gene and species trees. Three existing approaches are presented as based on: (a) tree-mapping, (b) annotating duplication, and (c) copying duplication modeling. Correspondences between (a) and (b), and (b) and (c) are mathematically explored. It is proven, in particular, that approaches (b) and (c) lead to equivalent duplication histories. Moreover, all the three approaches equivalently count the numbers of duplications and losses needed to explain all the differences between trees.

1. Introduction

It is today generally accepted that any two forms of life on earth have evolved from a common ancestor (J.M. Smith [**18**], Li and Graur [**13**]). One aim of evolutionary biology is the reconstruction of the evolutionary history of current species. Based on the assumption of common ancestors this history can be depicted as a tree, generally called a phylogenetic tree. Its nodes correspond to ancestral species and its edges are lines of descent.

The identities of species and the states of their various characters changed along the branches of the same evolutionary tree. Studying the history of a character is the main source of information for the reconstruction of evolutionary relationships among species. However, it is of prime importance to study characters that are based on evolutionarily comparable structures, called homologous. A fly and a bird both have wings and yet the bird is not more closely related to insects than to other vertebrates. The wings of birds and flies are believed to be incomparable structures. They seem unlikely to have evolved from the same structure in their most recent common ancestor. Estimating history from comparisons of structures non-comparable in this sense may lead to errors. With the rise of molecular biology the DNA sequences of genes have become available. These sequences provide DNA base pairs that can be treated as characters from which to estimate phylogenetic trees (see e.g. Fitch and Margoliash [**9**], Nei [**15**], Felsenstein [**8**]). However, determining which genes actually are comparable may be problematic. There exist large families

1991 *Mathematics Subject Classification.* 05C05, 92D15.

of related genes that have evolved through the processes of gene duplication. Once a gene has been duplicated, each copy can evolve distinct variations. Distinct copies of the same gene are called paralogous. Subsequently a single species may contain none, one, or several copies of what was a single gene in an ancestor. In order to derive a tree that correctly reflects the evolution of species of this particular family, one would like to know which copies of the gene are the comparable ones. Good estimates are generally only possible after careful study of the entire family. The tree derived from a selection of genes from a gene family and the tree describing the evolution of species will frequently have different topologies. We will call a tree describing the evolution of a set of genes the *gene tree* and the tree describing the evolution of species the *species tree.*

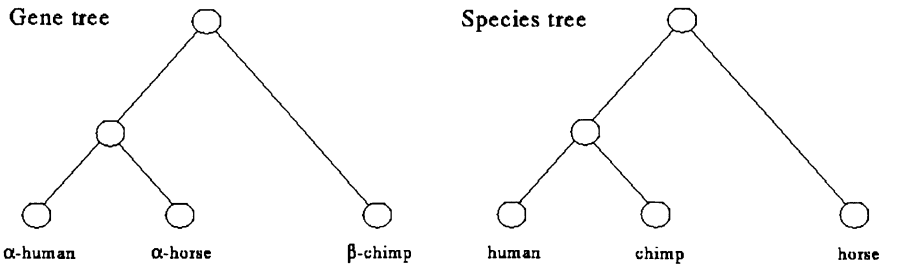


FIGURE 1. Incongruent gene and species trees.

Because they describe the evolution of different entities, a *gene tree* and *species tree* may be different in spite of the fact that their evolutionary representation is correct. Consider for example the *gene tree* and *species tree* for the hemoglobin family in Figure 1 (see, e.g. Li and Graur [**13**]). Goodman et al. [**10**] reasoned that this incongruence might result from mistaking paralogous genes for orthologous. The gene family of hemoglobin genes in vertebrates contains, among others, two types of genes: α-hemoglobin and β-hemoglobin. Both types evolved from an ancestral hemoglobin that existed prior to the vertebrates. This ancestral gene then was duplicated and the two new paralogous genes (copies) gave rise to vertebrate α- and β-hemoglobins, respectively. A researcher studying the α-hemoglobins from man, chimpanzee, and horse will find that man and chimpanzee have a common ancestor which in turn has a common ancestor with the horse. If the researcher studied β-hemoglobins from the same set of species he would find the same result. Were this family not as well-studied as it is today, the researcher might, however, have chosen a β-gene from chimp and α-genes from man and horse as the basis of his analysis. Consequently he would have found that man and horse group together with chimp of older evolutionary origin. This is believed to be correct for this particular selection of genes, but incorrect for the evolution of these species.

Figure 2 reflects the complete *gene tree* for the α-genes and the β-genes of man, chimp and horse. Note that the *gene tree* of Figure 1 is a subtree of the complete *gene tree* and the complete *gene tree* is a duplication of the *species tree* in Figure 1. Assume we would be aware of the duplication events in the *species tree.* Then we would be able to outline the topology of the *species tree* and to embed our *gene tree* into it. We would obtain a reconciled gene tree which represents in our case the complete *gene tree.* Thus, possible discrepancies between a *gene tree* and a *species*

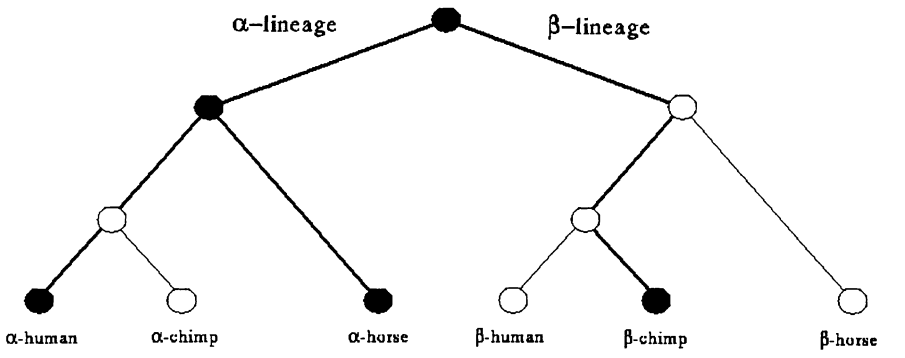


FIGURE 2. Complete gene tree.

tree can be explained by postulating duplication events that gave rise to different copies of a gene. From the theoretical point of view there is an unlimited number of sets of duplication events that result in a possible reconciled tree. Goodman et al. [**10**] outlined a strategy by postulating the duplications required for a reconciled tree. More recently this method has been elaborated in Page [**17**], Mirkin et al. [**14**] and Guigó et al. [**12**].

To introduce the basic logic we need to introduce gene and species trees in more detail. The basic assumption will be that exactly one gene from each contemporary species is present in the gene tree. Such an over-simplification is made for the sake of simplicity and, moreover, possibility of the resulting mathematical analysis. Besides, it is more a requirement to the representation of data rather than a restriction to the evolutionary processes covered. In the example considered, one should have two gene trees, one for hemoglobin α-lineage, the other for β-lineage, to compare each with a species tree. We may say that the term "gene" is used here in the meaning of molecular biology, as any distinct sequence variant, not a member of a prespecified paralogous family. In general, different genes should be treated within different gene families to yield gene trees satisfying the basic assumption (as it was unintentionally done by Guigó, Muchnik and Smith [**12**]). On the other hand, the assumption can be relaxed, which is however a subject for separate treatment.

Based on the assumption above, we may use the convention to denote a contemporary species and a contemporary gene from that species by the same symbol, which much simplifies the subsequent mathematical analysis. We will use integers for this purpose. Note that an ancestral gene is uniquely specified by the set of contemporary genes (leaves of the gene tree) descending from it. Likewise, an ancient species is uniquely specified by the contemporary species descending from it.

Duplication events are postulated based on a function, called the *tree mapping function.* This function maps each contemporary or ancestral gene of the gene tree onto a species in the species tree. This species, too, may either be contemporary, i.e. correspond to a leaf of the species tree, or ancestral, i.e. correspond to an inner node of the species tree. The tree mapping function maps a gene onto the most recent species that is presumed to have contained that gene. How to find out whether a species possessed a certain gene is easily seen from an example. Assume

that an ancestral gene has as contemporary descendant genes, 1, 2, and 3, and so call the ancestral gene $\{1,2,3\}$. Any species having 1, 2, and 3 among descendant species possessed a gene ancestral to genes 1, 2, and 3 and thus ancestral also to gene $\{1,2,3\}$. The most recent of these ancient species (called sometimes the *least common ancestor*) is the mapping image of $\{1,2,3\}$.

The tree mapping needs not be injective: it may map a parent gene, say a node a of the gene tree, onto the same species as one of a's immediate descendants called children, ca (we denote a fixed but arbitrarily chosen child of a node a with ca). This means that the most recent species which possessed gene a is also the most recent species in which one finds its child gene ca. In this case the bifurcation of a in the gene tree is not consistent with the bifurcation of its image in the species tree. The bifurcation of a takes place in the image and makes gene and species tree inconsistent with each other. In our model the bifurcation of a suggests that the species that is its mapping image possessed two copies of gene a, say $a+$ and $a-$. The existence of two copies is postulated to be due to a prior duplication of a predecessor gene, at what we call a *duplication node* in the species tree. Since we do not have knowledge of a species that possessed ca and not a we can identify ca with one copy, say $a+$. The sibling, denoted as $\bar{c}a$, of ca is the other copy $a-$ if $\bar{c}a$ maps onto the same species node as ca. If $\bar{c}a$ maps onto a species descendant to the duplication node, it is a descendant gene of $a-$. This distinction is the basis for distinguishing between two-side and one-side duplications below.

The number of duplications and other relevant events needed to explain a *gene tree* from a given *species tree* has been used as an asymmetric distance measure between the two trees by Goodman *et al* [**10**]. Among those events, the only one visible in terms of evolutionary trees is the loss of a certain set of genes (see, e.g., Nelson and Platnick [**16**], Page [**17**]). In our example those are the β-hemoglobin genes from man and horse and the α-hemoglobin gene from chimp. The lost genes might not constitute leaves but entire subtrees. Of course, the number of lost genes will grow with the number of duplications.

To determine the placement of gene duplication along the species tree we need to further elaborate on the inconsistencies between a gene and species trees caused by duplications and losses. We discuss three approaches to this: (a) tree-mapping, (b) annotating duplication, and (c) copying duplication.

Tree-mapping (a) has been considered by Guigò, Muchnik and Smith [**12**], whose basic idea is to measure the inconsistency, or mapping cost, of any duplication hypothesis by the sum over all genes of numbers of intermediate species nodes between the mapping image of a gene node and the image of its parent. No biologically meanigful motivation is given to this cost evaluation of the mapping which is claimed by the authors to count for all the loss events related to underlying duplications. This cost measure is minimized (with a local search algorithm) in the reconstructed evolutionary tree found in Guigò, Muchnik and Smith [**12**].

Annotating duplication (b) developed in Mirkin *et al* [**14**] (see also Eulenstein and Vingron [**7**] and L. Zhang [**19**]) involves a mathematical model of the duplication history corresponding to a duplication gene node g. All the leaves in the species tree whose corresponding genes belong to cg are annotated by $+$, and to $\bar{c}g$ by -. The pattern of the sign labels then naturally ascends in the species tree to the image of g. The maximum species nodes having all their content annotated by the same sign label correspond to the loss events.

Copying duplication (c), though never developed mathematically, is expressed quite clearly in the concept of reconciled tree (see Nelson and Platnick [**16**] and Page [**17**]). In this concept, the species subtree having a duplication node at its root is doubled so that one copy of the subtree keeps one copy of the duplicate gene g while the other subtree, another copy of the gene (as shown in Figure 2). In each of the tree copies, the gene is not observed in some of the species that are claimed to be "extinct". Being a two-dimensional graphical construction, the copying duplication becomes less clear when multiple and nested duplications of genes are hypothesized to explain the differences between a species and gene tree.

The purpose of this paper is to formally define all three approaches and to describe correspondences found among them. In particular, we prove that the number of loss events accounted by each of the approaches is the same. The paper integrates the earlier work of the authors exploring interconnections between approaches (a) and (b) (Eulenstein and Vingron [**7**], Eulenstein et al. [**5**]) and (b) and (c) (Eulenstein et al. [**6**]); proofs of some statements from those papers are omitted.

In Section 2, we formally define the notions just introduced and illustrate them with an example. Comparing approaches (a) and (b) is done in Section 3. In Section 4, approaches (b) and (c) are compared. Section 5 is a short conclusion.

2. Three Approaches to Comparing Gene and Species Trees

2.1. Basic Definitions. The model for evolutionary history we deal with is a rooted binary tree with the leaf set labeled by indices from an N-element set I. The indices label biological taxa under consideration and, simultaneously, genes one-to-one corresponding to the taxa. The basic assumption "one species - one gene" is the ground for such a twofold function of the indices, which simplifies mathematical formulas and derivations. If, for instance, a subset $g \subset I$ refers to genes and $s \subset I$ refers to species, expressions $s \subset g$, $s \cap g$, and $s \cup g$ are meaningful since both g and s are index subsets. Relation $s \subset g$ can be interpreted as "the genes corresponding to species in s all belong to g" or "the set of species corresponding to genes in g includes set s". The other set-theoretic expressions are interpreted similarly.

A *rooted tree*, T, is considered as a nested set of clusters, $T \subset 2^I$, which includes the singletons (leaves) $\{i\}, i \in I$, and I itself (the root). This implies that the terms "node of a tree" and "cluster in a tree" are considered synonymous in this paper. By $T(t)$ we denote the subtree of T rooted at $t \in T$; that is, $T(t) = \{t' \in T : t' \subseteq t\}$. An important property of a nested tree is that any two of its nodes are either nonoverlapping or nested (Estabrook and McMorris [**3**]).

A node $t \in T$ is internal if it is neither a singleton nor the root. The two children of an internal node $t \in T$ will be denoted by ct and $\bar{c}t$ (assigning c and $\bar{c}$ arbitrarily). The parent of a node $t \in T$ will be denoted by pt. For every subset $J \subset I$, the least common ancestor of J in T is minimum of the nodes $t \in T$ such that $J \subseteq t$. The least common ancestor of J will be denoted by $a_T(J)$.

A species tree will be denoted by S, with its clusters (nodes) $s \in S$, and a gene tree by G, with its clusters (nodes) $g \in G$. As explained above, both types of trees are subsets of 2^I.

Fig. 3 shows a species tree and gene tree. The letter c in the gene tree marks gene cluster $\{1, 2, 3\}$ and the letter F, in the species tree, marks species cluster $\{4, 5, 7\}$. Clusters G and 7 are children of F. Subtree $S(F)$ contains 4, 5, 7,

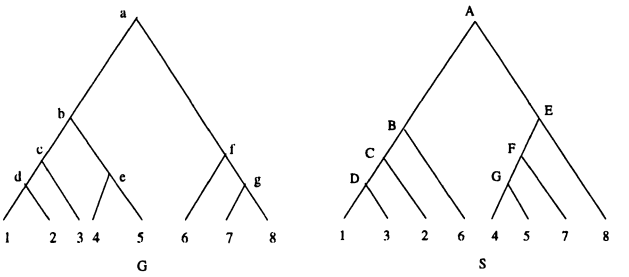


FIGURE 3. A gene tree, G, and a species tree, S.

$G = \{4, 5\}$ and F. For a subset $J = \{1, 2, 3, 4\}$ its least common ancestor is b in G and A in S.

2.2. Tree Mapping Modeling. The *tree mapping* function is just the least common ancestor mapping a_S in tree S applied to gene clusters $g \in G$: $a_S(g)$ is the minimum species cluster containing all genes from g.

A pair $(g, s) \in G \times S$ is called a *one-side duplication* if either $a_S(g) = a_S(cg)$ or $a_S(g) = a_S(\bar{c}g)$ (but not both). A pair $(g, s) \in G \times S$ is called a *two-side duplication* if both of the equations hold. A pair (g, s) is called a *duplication* if it is either a one-side or two-side duplication. The number of one-side duplications will be denoted $O(G, S)$. Sometimes, when there is no ambiguity, we refer to either s or g in a duplication pair (g, s) as a duplication, too.

Let us say that s is between s' and s'', or $s \in [s', s'']$ if $s' \subset s \subset s''$ $(s, s', s'' \in S)$. A node $s \in S$ will be called a g-intermediate if it is between $a_S(g)$ and $a_S(pg)$. The set of all g-intermediate nodes will be denoted I_g. Cardinalities of these sets can be employed as (local) measures of difference between trees G and S. In particular, the sets are empty if $G = S$. Let us refer to the total number of intermediate nodes in mapping G into S as to the *cost mapping* index, $M(G, S)$:

$$M(G, S) = \sum_{g \in G} |I_g| \tag{1}$$

This g-intermediate node concept is implicitly exploited in Guigó, Muchnik and Smith [**12**] where the following cost $C(g)$ (in comparing G and S) is assigned to every node $g \in G$: $C(g)$ equals $|I_{cg}| + |I_{\bar{c}g}|$, the total number of cg- and $\bar{c}g$-intermediate nodes in S (plus 1 when g is a one-side duplication). Their cost function $C(G, S)$ is defined as the sum of all costs $C(g)$, $g \in G$. (Actually, Guigó, Muchnik, and Smith [**12**] add to that the number of all the duplications; we drop this latter term since it is irrelevant in the present context.) Evidently, $C(G, S)$ equals the total number of the intermediate nodes in S plus the number of one-side duplications, as also observed by Eulenstein and Vingron [**7**] in their streamlining of the measures:

$$C(G, S) = M(G, S) + O(G, S) \tag{2}$$

To illustrate the tree-mapping concepts, let us consider the example of gene and species trees presented on Fig. 3. The mapping is shown in Fig. 4. It is readily seen that there are two duplications, at gene nodes a (two-side) and c (one-side), needed to explain all the differences between G and S. The images of all gene

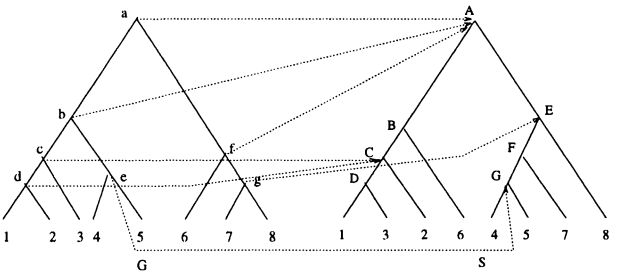


FIGURE 4. The least common ancestor mapping tree G into tree S from Fig. 3.

child-parent pairs are listed in Table 1. We can see that there are 7 intermediate nodes in total, which makes the cost function equal to 8.

TABLE 1. Mapping of the gene node-parent pairs into the species nodes.

G-node	Parent	Node S-image	Parent S-image	Number of Intermediates
1	d	1	C	1
2	d	2	C	0
3	c	3	C	1
4	e	4	G	0
5	e	5	G	0
6	f	6	A	1
7	g	7	E	1
8	g	8	E	0
g	f	E	A	0
f	a	A	A	0
e	b	G	A	2
d	c	C	C	0
c	b	C	A	1
b	a	A	A	0

2.3. Annotating Duplication Modeling. Any duplication node marks a difference in patterns of divergence between the species tree and the gene tree. The difference is readily seen in comparing the children of a duplication pair (g, s). It is characterized by the following "inconsistency" condition (Mirkin et al. [**14**]):

$$g \subseteq s \text{ and } cg \cap cs \neq \emptyset \text{ and } cg \cap \bar{c}s \neq \emptyset \tag{3}$$

for an appropriate denotation of children of g. Such an inconsistency requires a duplication of the ancestral gene g postulated in the ancestral species s so that all species containing genes descendant from gene cg carry one copy of the duplicate gene and all species containing genes descendant from gene $\bar{c}g$ contain the other copy (Duplication/Speciation Principle in Mirkin, Muchnik and Smith [**14**]).

A mapping $\delta : S(s) \to \{+/-, +, -, \emptyset\}$ is referred to as an *annotating duplication* if and only if for any $s' \in S(s)$,

$\delta(s') = \emptyset$ if both of the two following conditions hold: (a) $s' \cap cg = \emptyset$ and (b) $s' \cap \bar{c}g = \emptyset$;

$\delta(s')$ is $+$ or $-$ if either of (a) or (b) holds, and
$\delta(s') = +/-$ if none of (a) and (b) holds.

The mapping δ indicates whether one, the other, both or neither duplicate gene is present at each node of the species tree (see Fig. 5).

The evolutionary history of the duplication generated by an inconsistent pair (g, s), where $s = s_S(g)$, can be considered in the framework of the basic partition $\{cg, \bar{c}g\}$ of g put into the context of species subtree $S(a_S(g))$. The elements of each of the classes, cg or $\bar{c}g$, are interpreted as the currently living species bearing only one of the copies of the duplicated gene g. Due to the definition of annotating δ above any node $s \in S(a_S(g))$ can be qualified as *one-copy* ($+$ or $-$ only) if $g \cap s$ is included in one of the classes only or *mixed* (and labeled by $+/-$ mark) if s overlaps both of them. A particular evolutionary meaning is assigned to maximal one-copy nodes $s \in S(a_S(g))$: each of them corresponds to the event of *loss* of a duplicate copy.

Explicitly, the concept of loss can be formulated as follows. A node $s \in S$ will be referred to as a g-loss if and only if any of the two equivalent statements is true:

(i) $s \cap g \subseteq c\bar{g}$ or $s \cap g \subseteq \bar{c}g$, but this is not true for its parent, ps;

(ii) $s \cap cg = \emptyset$ or $s \cap \bar{c}g = \emptyset$, but this is not true for its parent, ps.

In the case when both of the inclusions in (i), or equations in (ii), are satisfied, that is, if $s \cap g = \emptyset$, there is no information to assign a copy of the duplication to s; this corresponds to what is called *gap* in Mirkin, Muchnik and Smith [**14**]. It should be noted however that, in the latter paper, the concept of gap is considered somewhat ambigously so that the current meaning corresponds to that based on the Duplication/speciation principle (p. 500) while that introduced on p. 498 may refer not only to the losses, but some smaller nodes, too.

A g-loss s that is not gapped (so that $s \cap g \neq \emptyset$) will be referred to as a charged loss.

As the gene and species trees presented in Fig. 4 give two duplication pairs, (a, A) and (c, C), corresponding annotating duplications are shown in Fig. 5. The losses are shown with boxes in the copies of the species tree. Species from different children of a duplicate genes are labeled by different sign labels, $+$ or $-$, according to annotating duplication. We can see that there are 5 losses for (a, A) and 3 losses for (c, C), which gives 8 losses in total, which was the same counted by the cost function above.

Guigó, Muchnik and Smith [**12**] defined that $C(G, S)$ in (2) was always exactly the total number of losses in duplication-based comparing G and S, which was conjectured by Mirkin, Muchnik and Smith [**14**] with regard to the losses defined in terms of the latter's concept of (annotating) duplication. The conjecture has been differently proved by Eulenstein and Vingron [**7**], Zhang [**19**], and Eulenstein, Mirkin and Vingron [**5**] (see Section 3 below).

2.4. Copying/Reconciling Duplication Modeling. In the previous account, the duplications have been considered as virtual ones: the species evolutionary tree was not affected, but just annotated with particular gene histories. However, one might also think of a duplication $s \in S$ as of really occurred in the evolution, as that presented in Fig.2, and thus requiring corresponding change of the species tree. Such a change assumes that the subtree $S(s)$ is doubled in S so that one copy of the subtree keeps one copy of the duplicate gene g while the other

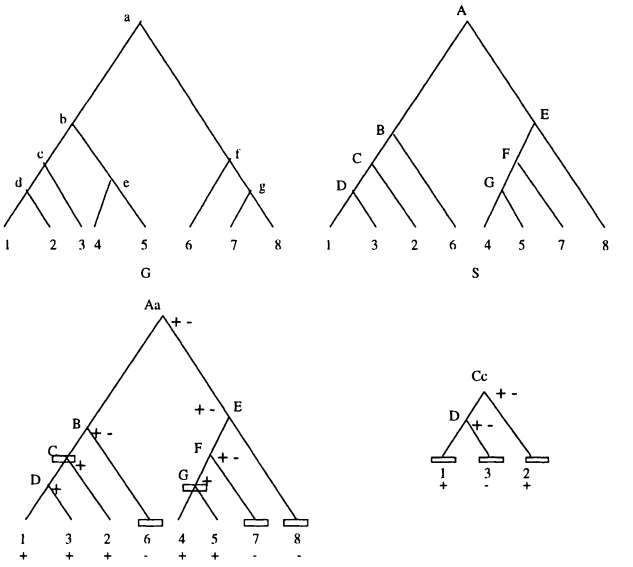


FIGURE 5. The losses for two duplications, Aa and Cc, presented by boxes in sign labeled copies of species subtrees.

subtree, another copy of the gene. Such a transformation of S into a "reconciled" tree has been developed by natural historians (see Page [**17**] for references) to reconcile the information in tree S with that in tree G.

The definition of reconciled tree presented by Page in [**17**], p. 63-64, is based on the concept of multiset tree containing any species node as many times as needed. Evidently, care must be taken to label the duplicates accurately, which has not always been done in [**17**] and earlier works.

To give an appropriate mathematical definition to the concept of copying duplication, suppose a duplication of gene G occurred during the evolution of $s \in S$, just before the ancestral species s appeared. The duplication is conceived as two copies of subtree $S(s)$, each labeled by the corresponding copy, c_1 or c_2, of the gene. This means that every node $s' \in S(s)$ is recoded as $s'c_1$ in one copy of $S(s)$ and $s'c_2$ in the other copy. The copies are denoted as $S(s)c_1$ and $S(s)c_2$. In each of the trees, $S(s)c_1$ and $S(s)c_2$, some of the leaves correspond to species in I and some do not. The first will be called active, the second non-active (extinct, in terminology of Page [**17**]). Let us denote the set of active species $i \in s$ in $S(s)c_k$ by $c_k(s)$ $(k = 1, 2)$. In principle, $c_1(s) \cap c_2(s) \neq \emptyset$ since both of the gene copies can be present in currently living species. However, in the context of present paper, we analyze the case when only one of the diverged gene copies is present in the material under consideration. This requirement is a prerequisite to defining the concept of annotating duplication above. Therefore, we postulate here that the active species sets are not overlapping: $c_1(s) \cap c_2(s) = \emptyset$. At the same time, there is no need that every species in s is active; that is, there can be $s - (c_1(s) \cup c_2(s)) \neq \emptyset$ so that no information on the copy c_1 or c_2 in $i \in s - (c_1(s) \cup c_2(s))$ is available. Finally, let us denote by $S(s)\{c_1, c_2\}$ the labeled rooted tree consisting of two subtrees, $S(s)c_1$ and $S(s)c_2$, whose roots, sc_1 and sc_2 are children of the formal root labeled by s,

not necessarily a subset of I. This time, it is the union of all species in s copied, that is, the cluster corresponding to s consists of new, relabeled leaves, ic_1 and ic_2, for all $i \in s$. The tree $S(s)\{c_1, c_2\}$ along with the active sets $c_1(s)$ and $c_2(s)$ is a graph-theoretic model of the concept of copying duplication. Note that all the nodes of $S(s)\{c_1, c_2\}$ are labeled, in contrast to the original gene and species trees that are considered leaf-labeled only.

The maximal subsets of non-active leaves in $S(s)\{c_1, c_2\}$ can be interpreted as losses of the corresponding genetic material (see Page [**17**]), which leads to the following definition. Let us consider any node $sc \in S(s)\{c_1, c_2\}$ as a cluster consisting of the leaves in the corresponding subtree. Then, a subset $t \in I$ will be referred to as a *species cluster* if there exists a node $sc \in S(s)\{c_1, c_2\}$ such that $i \in t$ if and only if $ic \in sc$. A maximal species cluster consisting of the non-active copies will be referred to as a *-loss.

In Fig. 6, a copying duplication in the species tree root, A, is present as corresponding to the duplication pair Aa. The labels of children of a in G, b and f, are exploited as the duplicate copy labels. The leaves in corresponding children b and f clusters are marked by a as active ones; the other, non-active, leaves by star; *-losses are shown by boxes.

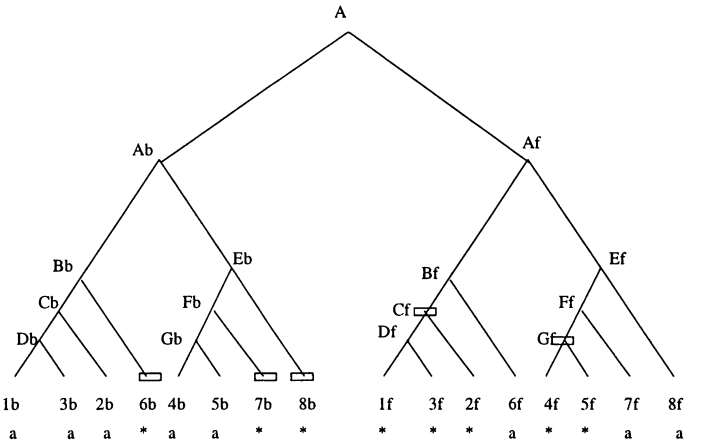


FIGURE 6. A copying duplication tree for duplication pair Aa in Fig. 5; b and f, the labels of children of a in G, are used as the duplicate labels.

In terms of the species tree S, implementation of a copying gene duplication in its node $s \in S$ is by substitution of the subtree $S(s)$ with the duplicated tree $S(s)\{c_1, c_2\}$. Other duplications in $S(s)$ can be implemented recursively in either subtree, $S(s)c_1$ or $S(s)c_2$, by adding new labels to those of previous duplications. The active species sets of these subsequent duplications must be subsets of the previous active sets. Let us formulate a "labeled" version of the reconciling algorithm from Page [**17**] implementing this construction.

Algorithm for Constructing Labeled Reconciled Tree

The algorithm recursively updates the tree R under processing by observing pairs $(g, r) \in G \times R$ in their natural partial order:

Initially, put $R = S$ and the natural partial order being just set-theoretic inclusion on pairs $(g, s) \in G \times S$.

Any time when (g, r) is a duplication, replace the subtree $R(r)$ with the copying duplication subtree, $R(r)\{cg, \bar{c}g\}$, consisting of two copies of $R(r)$ having r as their parent. To all the node labels in one of the copies is added the symbol cg; to the nodes in the other copy the symbol $\bar{c}g$ is added. In the copy labeled cg, all the leaves $i \notin cg$ are labeled non-active with "*"; in the other copy, label the leaves $i \notin \bar{c}g$ with "*".

Update the natural partial order to include those pairs (g, r) whose label suffixes contain a node above g in G.
Go to the next duplication pair (g, r); end, if there is no duplication pair left.

The final output is the labeled reconciled tree $R(G, S)$.

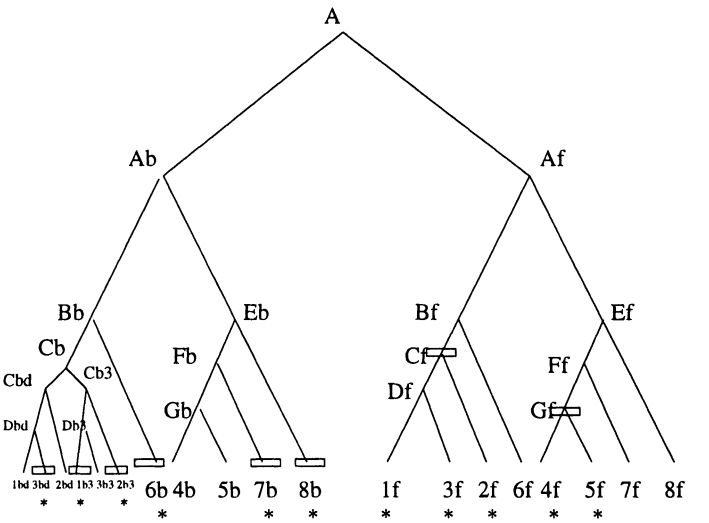


FIGURE 7. The reconciled labeled tree $R(G, S)$.

To illustrate the algorithm, let us apply it to G and S in Fig. 4. Since there are only two duplication pairs, (a, A) and (c, C), there are only two copying iterations needed. Tree R resulting from S after the first iteration is, actually, that shown in Fig. 6. The final reconciled tree is drawn in Fig. 7. We can see that the number of *-losses is again 8. Moreover, they obviously correspond to the losses in Fig. 5.

In Eulenstein, Mirkin and Vingron [**6**], a general concept of the joint duplication history tree is introduced. The questions of correctness of the algorithm above

(impossibility of cycles) and correspondence between losses and *-losses are also addressed in that paper, which will be reviewed in Section 4.

3. Correspondence Between Losses and Intermediates

3.1. Some Structural Properties of the Losses. The set of all g-losses, for a $g \in G$ given, will be denoted by L_g; it has a fairly simple structure.

STATEMENT 1. *For any $g \in G$, L_g is a partition of $a_S(g)$ whose restriction to g fits strictly within the basic partition $\{cg, \bar{c}g\}$.*

Proof: Indeed, the losses are nodes in S and, thus, must not overlap each other (since they may not be nested by the requirement of their maximality). On the other hand, any leaf $i \in a_S(g)$ belongs to a g-loss: if not, $\{i\}$ is a g-loss on its own. The fact that $g \cap s$ fits within partition $\{cg, \bar{c}g\}$ for any g-loss s follows directly from the definition. Moreover, it cannot be only two g-losses because there is no inconsistency in such a case. □

It follows, from the statement, that the minimum number of g-losses, for any g, is 3. The only case when it is possible is that g is a one-side duplication and that one of the images $a_S(cg)$ and $a_S(\bar{c}g)$ which is strictly included in $a_S(g)$ has both of its children being g-losses.

Particular attention will be paid to the losses that are children of the corresponding duplication nodes.

STATEMENT 2. *A child, cs, of a duplication node $s \in S$ is a g-loss if and only if (g, s) is a one-side duplication; that is, $a_S(\bar{c}g) \subset s = a_S(g) = a_S(cg)$ (under appropriate c and $\bar{c}$ labeling), and $cs \cap a_S(\bar{c}g) = \emptyset$.*

Proof: Let us show, initially, that if cs overlaps $a_S(\bar{c}g)$ then it is not a g-loss since it overlaps both cg and $\bar{c}g$. Indeed, the fact of overlapping means that $a_S(\bar{c}g) \subseteq cs$, which implies $\bar{c}g \cap cs \neq \emptyset$. The other condition, $cg \cap cs \neq \emptyset$ follows from the fact that $s = a_S(cg)$, which means that cg must overlap both children of s.

If cs does not overlap $a_S(\bar{c}g)$, then it does not overlap $\bar{c}g$ either, which is not true for its parent s. □

It follows from statement 2 that the total number of the "child" losses (each being a child of a duplication) is equal to the number of one-side duplications, $O(G, S)$.

The property proven implies also that no child of a two-side duplication (g, s) can be a g-loss, thus both must be mixed nodes.

3.2. Principal Correspondence. Let us consider an arbitrary $s \in S$ and denote the set of duplications (or, those one-side duplications) $g \in G$ for which s is a g-loss (or, a g-loss being a child of $a_S(g)$) by D_s (or, by O_s). Obviously, $O_s \subseteq D_s$. Let us denote by P_s the set of all nodes g for which ps is a g-intermediate while s is its collateral child. The latter denotation means that $g \in P_s$ if and only if the parent of s, ps, belongs in the path connecting $a_S(g)$ and $a_S(pg)$ in S so that $ps \in [a_S(g), a_S(pg)]$ along with the other child, s', of ps (not s) also belonging to the path as presented in Fig. 8 where the trees, G and S, are drawn as just triangles. While ps must be between the images, $a_S(g)$ and $a_S(pg)$, coinciding with neither of them, the sibling of s, s', may coincide with $a_S(g)$.

We are going to prove that these sets are interelated so that

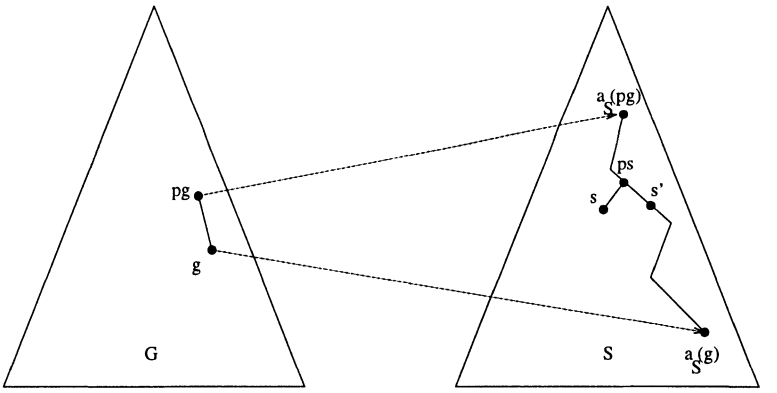


FIGURE 8. The set P_s consists of nodes $g \in G$ satisfying the configuration concerning path between $a_S(g)$ and $a_S(pg)$ in S presented.

$$|D_s| = |P_s| + |O_s| \tag{4}$$

Summing up equations in (4) by all $s \in S$, we will get the following result.

STATEMENT 3. *The total number of losses, $L(G,S)$, is equal to the total number of the mapping intermediate nodes plus the number of one-side duplications:*

$$L(G,S) = M(G,S) + O(G,S) \tag{5}$$

This statement proves that the number of losses $L(G,S)$ is exactly the cost function, $C(G,S)$, introduced by Guigó et al. in [**12**], which gives a correspondence between the tree-mapping and annotating approaches. However, this correspondence relates only to the numbers: the intermediate nodes are not, and cannot, be the losses. An interpretation of the intermediate nodes following from (4) is discussed below in section 3.4.

3.3. A Technical Proof. In Eulenstein and Vingron [**7**] equation (4) is considered in the form $|D_s| - |O_s| = |P_s|$. The equation is stated as a main result (Theorem 4.2), though in a different notation. An edge, (s, ps), is denoted by e while $L^*(e)$ denotes subset P_s and $C^*(e)$ is used to denote subset $D_s - O_s$.

Then, $L^*(e)$ is partitioned into two parts, $L^*_=(e)$, being defined as those elements of $L^*(e)$ also in $C^*(e)$, and $L^*_{\neq}(e)$, the rest. Similarly, $C^*(e)$ is partitioned into $C^*_=(e)$ and $C^*_{\neq}(e)$. The Venn diagram of these subsets in the set of all gene tree nodes is shown in Fig. 9.

After this, the essential statement becomes $|L^*_{\neq}(e)| = |C^*_{\neq}(e)|$ which is proven in Eulenstein and Vingron [**7**] by, first, observing that $L^*_{\neq}(e)$ is an antichain (none of its elements is a part of another one) while every node in $C^*_{\neq}(e)$ contains a node from $L^*_{\neq}(e)$, and, second, analyzing the properties of the tree obtained from G by cutting off all the nodes descending from nodes in $L^*_{\neq}(e)$.

A problem with the proof outlined above is that it is rather technical and gives no biologically interpreted correspondence between sets $L^*(e) = D_s - O_s$ and

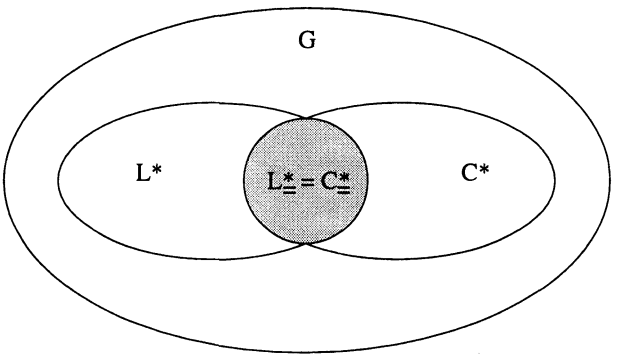


FIGURE 9. Venn diagram of sets $L^*(e)$ and $C^*(e)$.

$C^*(e) = P_s$ where $e = (s, ps)$. In the paper by Eulenstein et al. [5], such an interpretational proof has been provided, as follows.

3.4. An Interpretative Proof. Let us fix an interior $s \in S$ and consider all those duplication pairs $(g, a_S(g))$ that satisfy the following conditions: (i) s is not a child of $a_S(g)$, and (ii) s is a g-loss. A typical pattern is shown in Fig. 10. Now, let us fix an inclusion-maximal g from the set of duplications satisfying (i) and (ii) to deal with it in the rest of this section. It can happen that, for an s given, there is no such duplications at all. The following relates to the case when duplications satisfying (i) and (ii) exist.

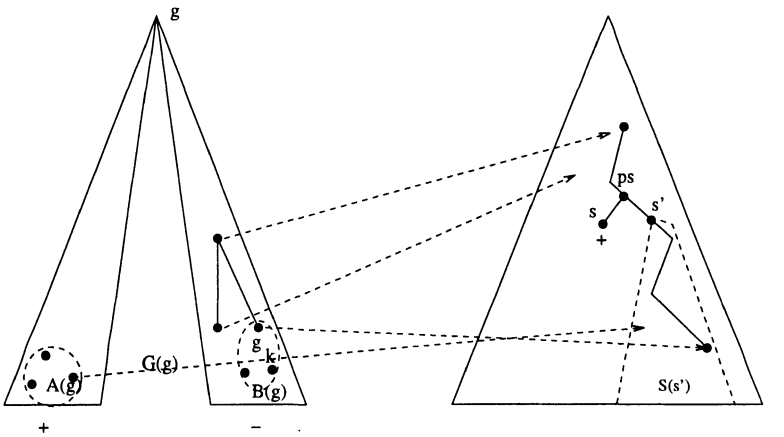


FIGURE 10. The structure of $Q(g)$ with respect to s and its sibling s'.

It can be proven that, since g under consideration is maximal, s is a charged (not gapped) g-loss so that, say $g \cap s \subseteq cg$ [5]; in Fig. 10 this loss is labeled by plus. Let us consider subtree $G(g)$ and define nodes $g_1, ..., g_q \in G(g)$ that are maximal parts of some nodes in $S(s')$ (s' is the collateral child of ps, see Fig. 10)

while their siblings are not parts of s. Let us denote the set of these nodes by $Q(g) = \{g_1, ..., g_q\}$. Obviously, none of the nodes $g_k \in Q(g)$ overlaps s since they are parts of its sibling.

STATEMENT 4. *For any $g_k \in Q(g)$, ps is a g_k-intermediate while s is the collateral child. Moreover, no other node $\bar{g} \in G(g)$ exists so that ps is a $\bar{g}$-intermediate and s is the collateral child.*

Proof: Indeed, $a_S(g_k) \subset ps$ because, as was defined above, $g_k \subseteq s'$ where s' is the sibling of s. On the other hand, $ps \subset a_S(pg_k)$ since pg_k is not included in $ps = s \cup s'$ (by definition of g_k) though overlaps it. Node s is the collateral child since $g_k \subseteq s'$.

Then, for ps to be an intermediate along with s the collateral child, for a pair $(\bar{g}, p\bar{g})$, $p\bar{g} \in G(g)$, it must be $\bar{g} \subseteq s'$ where s' is the other sibling of s, while $p\bar{g} \not\subseteq ps$. That implies that $\bar{g}$ must be a maximal node to be a part of a node in $S(s')$, but its sibling is not a part of s. Thus, $\bar{g}$ has been counted in $Q(g)$. □

Thus, we have proven that

$$Q(g) = P_s \cap G(g) \tag{6}$$

Now, we are going to explore whether different duplications in $(D_s - O_s) \cap G(g)$ can be assigned to different elements in $Q(g)$. It is expected that the answer is yes, which can be exploited in proving equation (4).

A one-to-one assignment of the duplications to elements of $Q(g)$ can be done with a procedure involving the following concepts. Let us partition the nodes in $Q(g)$ into two classes by charge: $A(g)$ - those charged as s; and $B(g)$ - those charged alternately. An important thing is that $B(g)$ cannot be empty since ps must be mixed. Thus, $q = |A(g)| + |B(g)|$ and $|B(g)| > 0$. It is not difficult to establish a one-to-one duplication assignment for the nodes in $B(g)$ and then to move on considering the other part of $Q(g)$, which can be exploited in a recursive manner: at each step, a non-empty subset of the remaining part of $Q(g)$ is separated to make an easy duplication assignment and then move on to the rest, until no nonassigned elements in $Q(g)$ remain. The procedure can be described as follows.

One-to-One Assigning Duplications from $(D_s - O_s) \cap G(g)$ to Elements of $Q(g)$

0. Initial setting: $Q \leftarrow Q(g)$.

1. Find $|B(g)|$ different duplications for the nodes in $B(g)$ and extract $B(g)$ from Q: $Q \leftarrow Q - B(g)$ along with $q \leftarrow |Q|$. This is done via considering the least common ancestors, $a_S(g_1, g_2)$, for every pair $g_1, g_2 \in B(g)$. All of them are right duplications as stated in the Statement 5 below. By the properties of binary trees, exactly $|B(g)| - 1$ of them are different, which, together with the top duplication g, gives $|B(g)|$ different duplications.

2. Check whether $Q = \emptyset$. If yes, end. If not, go to 3.

3. Find a $g_k \in Q$ and a corresponding duplication $\bar{g}_k$ along with the corresponding set $Q(\bar{g}_k) \subseteq Q$ such that $B(\bar{g}_k) \neq \emptyset$ in the corresponding partition of $Q(\bar{g}_k)$ by charge. [For any g_k, duplication $\bar{g}_k$ is defined as the minimal ancestor of g_k (in G) to overlap s, which is justified by Statement 6 below. By its very definition, $\bar{g}_k$ belongs to the part of $G(g)$ charged as s and, thus, is different from the duplications assigned at step 1.] Since all the $\bar{g}_k$s are partially ordered by inclusion, pick a maximal one; its $B(\bar{g}_k) \neq \emptyset$, obviously. Go to step 1 with $g = \bar{g}_k$.

The procedure obviously converges since Q is decreased at every step. Correctness of step 1 in the procedure is proved by the following.

STATEMENT 5. *For any pair of nodes in $B(g)$, their minimal common ancestor in G is a duplication, in $G(\bar{c}g)$, such that s is its non-child loss.*

Proof: Let $\bar{g} = a_G(g_1 \cup g_2)$, then, obviously, pg_1 and pg_2 are among the nodes in $G(\bar{g})$. This implies that the images of $\bar{g}$ and each of its children, $c\bar{g}$ and $\bar{c}\bar{g}$, include ps and, thus, are nested. Let $a_S(c\bar{g}) \subseteq a_S(\bar{c}\bar{g})$, then both, $c\bar{g}$ and $\bar{c}\bar{g}$, are parts of $a_S(\bar{c}\bar{g})$, which means that $a_S(\bar{g}) = a_S(\bar{c}\bar{g})$ and, thus, $\bar{g}$ is a duplication. Since both g_1 and g_2 are from $\bar{c}g$, so is $\bar{g}$, which proves that both $\bar{g} \in G(\bar{c}g)$ and $s \cap \bar{g} = \emptyset$. However ps includes both g_1 and g_2 thus overlapping both children of $\bar{g}$, which means that s is a $\bar{g}$-loss (gapped). The fact that s is not a child of $a_S(\bar{g})$ follows from inclusion $ps \subset a_S(\bar{g})$. □

Correctness of step 3 in the procedure is proved by the following.

STATEMENT 6. *For any $g_k \in A(g)$, its minimal ancestor $\bar{g}_k$ overlapping s is a duplication, in $G(cg)$, for which s is a non-child loss.*

Proof: The fact that such a $\bar{g}_k \in G(cg)$ exists follows from $g_k \in G(cg)$ and the assumption that $s \cap cg \neq \emptyset$. There are two cases possible, $pg_k = \bar{g}_k$ and $pg_k \subset \bar{g}_k$, which will be considered in turn.

First, let $pg_k = \bar{g}_k$ so that $pg_k \cap s \neq \emptyset$ and, thus, the other sibling, g'_k, of g_k overlaps s, too, since g_k and s are not overlapping. Let us show that $a_S(pg_k) = a_S(g'_k)$, that is, $\bar{g}_k = pg_k$ is a one-side duplication. Indeed, if $a_S(g'_k) \subset a_S(pg_k)$, then $a_S(g_k) \cap a_S(g'_k) = \emptyset$ and thus g'_k may not overlap anything in the path between $a_S(g_k)$ and $a_S(pg_k)$, ps included, which is not true. The fact that s is a non-child pg_k-loss follows from that ps overlaps both children of $a_S(pg_k)$ but ps does not coincide with $a_S(pg_k)$.

Second, let $pg_k \subset \bar{g}_k$ so that pg_k is a part of a child of $\bar{g}_k$, say, $pg_k \subseteq c\bar{g}_k$. Then $g_k \subset c\bar{g}_k$ so that $s \cap \bar{c}\bar{g}_k \neq \emptyset$ since, otherwise, $s \cap c\bar{g}_k \neq \emptyset$ which contradicts the minimality of $\bar{g}_k$. Thus, s and $a_S(\bar{c}\bar{g}_k)$ are overlapping and so do $a_S(\bar{c}\bar{g}_k)$ and $a_S(c\bar{g}_k)$ because $s \subset ps \subset a_S(pg_k) \subseteq a_S(c\bar{g}_k)$. This means that the latter two sets are nested so that, for instance, $a_S(\bar{c}\bar{g}_k) \subseteq a_S(c\bar{g}_k)$. Thus, both $\bar{c}\bar{g}_k$ and $c\bar{g}_k$ are included in $a_S(c\bar{g}_k)$, which implies $\bar{g}_k = \bar{c}\bar{g}_k \cup c\bar{g}_k$ is included in $a_S(c\bar{g}_k)$ and, therefore, $a_S(\bar{g}_k) = a_S(c\bar{g}_k)$, that is, $\bar{g}_k$ is a duplication indeed. Node s is $\bar{g}_k$ one-copy since $s \cap c\bar{g}_k = \emptyset$. However, ps includes g_k and thus overlaps $c\bar{g}_k$, which proves that s is a $\bar{g}_k$-loss (not being a child of $\bar{g}_k$ since $ps \subset a_S(\bar{g}_k)$). □

STATEMENT 7. *The cardinalities of $P_s \cap G(g)$ and $(D_s - O_s) \cap G(g)$ coincide:*

$$|P_s \cap G(g)| = |(D_s - O_s) \cap G(g)| \tag{7}$$

Proof: The fact that the left part is not greater then the right follows from correctness of the assignment procedure because equation (6) holds. On the other hand, $g' \in (D_s - O_s) \cap G(g)$ implies that s is a g'-loss and its parent ps is between the images of g' and a child, say cg', $ps \in [a_S(cg'), a_S(g')]$, which means that $cg' \in P_s \cap G(g)$. This proves that the right part in (7) is not greater than the left. □

STATEMENT 8. *For any node* $s \in S$,

$$|P_s| = |D_s| - |O_s| \tag{8}$$

Proof: Since maximal duplications g are nonoverlapping, so are corresponding subtrees, $G(g)$. Thus, summing up all the equations (7) gives (8). □

This completes the proof of statement 3 since (8) is equivalent to (5).

4. Equivalence Between Annotating and Copying Duplications

4.1. The Structure of Duplication Nodes in G. Let us consider the set of all duplication nodes in S. These nodes along with edges representing set theoretic inclusion as in a Hasse diagram (for any node, only the nodes being its "immediate" subsets are its children) form a forest. Since there is no overlap between the species in different connected components of the forest, each of the components can be considered independently. Let S^* be such a component and G^* be its corresponding counterpart in G; that is, $g \in G^*$ if and only if (g, s) is a duplication at some $s \in S^*$.

STATEMENT 9. *The set* G^* *is a connected graph (by set-theoretic inclusion), and thus a rooted tree.*

Proof: Let (g_1, s_1) and (g_2, s_2) be duplications such that s_1 and s_2 are nodes in S^* connected by a path so that one of them is a part of the other, for instance, $s_1 \subseteq s_2$. If $g_1 \subseteq g_2$, then g_1 and g_2 are obviously connected in G^*. If $g_1 \cap g_2 = \emptyset$, then let us consider their least common ancestor in G, g, and prove that $g \in G^*$ which implies g_1 and g_2 are connected in this case, too. If both g_1, g_2 were parts of cg (or $\bar{c}g$), then cg (or $\bar{c}g$), not g, would have been the least common ancestor, which shows that $g_1 \subseteq cg$ and $g_2 \subseteq \bar{c}g$. So $g_1 \subseteq a_S(cg)$ and $g_2 \subseteq a_S(\bar{c}g)$, which implies $s_1 \subseteq a_S(cg)$ and $s_2 \subseteq a_S(\bar{c}g)$. Thus, $a_S(cg) \cap a_S(\bar{c}g) \neq \emptyset$, i.e. one of the sets is a part of the other, say $a_S(cg) \subseteq a_S(\bar{c}g)$. Thus $a_S(g) = a_S(\bar{c}g)$, and so $(g, a_S(g))$ is a duplication with $a_S(g)$ obviously belonging to S^*.

It remains to prove now that g_1 and g_2 are also connected for nonoverlapping $s_1, s_2 \in S^*$. The immediate predecessor, s, of s_1 and s_2 in S^* corresponds to a vertex $g \in G^*$ so that pairs $\{g_1, g\}$ and $\{g_2, g\}$ are connected in G^* which proves that g_1 and g_2 are connected. □

In the remainder, we restrict ourselves to the case when S^* is a connected component, because the results can be easily extended to the general case when S^* is a forest.

The set of the nodes of tree G^* can be partitioned into classes of nodes mapped into the same image $s \in S^*$. Obviously, these classes are "convex" parts of G^*: if g_1 and g_2 belong to such a class and there exists a $g_3 \in G^*$ such that $g_1 \subset g_3 \subset g_2$, then g_3 belongs to the same class. Indeed, $a_S(g_1) = a_S(g_2) = s$ implies $a_S(g) = s$ for any gene tree node between g_1 and g_2. This property allows us to partition the tree G^* into rooted subtrees (within the mapping classes), $G^*(g, s)$, where s is the common image of all the nodes in $G^*(g, s)$ and g is its root. The subtrees $G^*(g, s)$

are supposed to be maximal, that is, any leaf of a $G^*(g,s)$ has its children in G^* mapped into node(s) different from s. This implies that any leaf of a $G^*(g,s)$ has at least one of its G-children out of G^*. Indeed, by the very definition of a duplication node, one or both of its children must have the same mapping image and, thus, belong to $G^*(g,s)$ if this one or both belong to G^*.

Let us denote by G^{**} the set-inclusion tree obtained from G^* by adding to it all G-children of its nodes (compare G^* with G^{**} in Fig. 11). Each of the subtrees, $G^*(g,s)$, of G^* also will be extended in G^{**} by adding all G-children of its nodes. The extended $G^*(g,s)$ will be denoted by $G^{**}(g,s)$. Obviously, any $G^{**}(g,s)$ is a subtree in G^{**}. According to this construction, among the leaves, $g_1,...,g_m$, of a subtree $G^{**}(g,s)$ there may occur both nonduplication and duplication nodes. Any leaf, g_k, which is a duplication node, is the root of another subtree $G^{**}(g_k,s_k)$. Any leaf that is not a duplication node is either a leaf of G^{**} or the parent of another subtree $G^{**}(g',s')$. Obviously, the leaves, $g_1,...,g_m$, of $G^{**}(g,s)$ considered as clusters in I form a partition of the cluster g. The number of the leaves, m, is the number of duplication nodes in $G^{**}(g,s)$ plus 1.

In Fig. 11, the two subtrees $G^{**}(g,s)$ in G^{**} are separated by the dashed boxes.

4.2. Plait Frame and Plait Tree. Let us define a concept of *plait frame*, $P(G^*,S^*)$, which is a labeled rooted binary tree following, in general, the pattern of tree G^{**}. However, insertions from S are made between different subtrees $G^{**}(g,s)$.

To define $P(G^*,S^*)$, take G^{**} and consider all situations when a subtree, $G^{**}(g,s)$, is incident (from below) to another subtree, $G^{**}(g',s')$; that is, $s \subset s'$, and g is either a leaf of $G^{**}(g',s')$ or a child of a leaf, $\bar{g}$, of $G^{**}(g',s')$. In the former case, the parent of g in G^{**} is a node $g'' \in G^*(g',s')$. In the latter, the grandparent of g (parent of $\bar{g}$) is a node $g'' \in G^*(g',s')$. Let us denote by $S(s,s')$ a subtree of S consisting of the path between s and s' in S (s' included, s excluded) along with the collateral children of the vertices in the path added. This subtree must be inserted between g and the g'' just defined. When a leaf, $\bar{g}$, of $G^{**}(g',s')$ is g's parent, the edge $(g,\bar{g})$ is substituted by the subtree $S(s,s')$ so that the parent of s in $S(s,s')$ becomes the parent of g. When there is no node between g and g'' (the former case), a node $\bar{g}$ is inserted in the edge (g,g') artificially just to be substituted by the subtree $S(s,s')$ as above. After all insertions are made, the structure of the plait tree, $P(G^*,S^*)$, is defined. It remains to label all nodes in $P(G^*,S^*)$.

In Fig. 11 node b is a leaf parent node (the latter case) replaced by the path AB from S along with collateral children, E of A and 6 of B, in $P(G^*,S^*)$.

The nodes of $P(G^*,S^*)$ are labeled by words of form sw where $s \in S^*$ and suffix $w \in W$; W stands for the set of finite words over alphabet G^{**} (that is, its letters are nodes of G^{**}) containing not more than one occurrence of each of the letters; moreover, the length of $w \in W$ is not greater than the depth of tree G^*. The labeling is defined differently for the two types of nodes in $P(G^*,S^*)$: those inherited from G^{**} and those inserted from S. There are thus two rules:

(a) Each node $g' \in G^{**}(g,s)$ is labeled by swg' where $w \in W$ is the suffix of the label assigned to its parent in $P(G^*,S^*)$ (so that w is empty at the root of G^{**}).

(b) Each node $s'' \in S(s,s')$ carries the label $s''w$ where suffix w is the same as that in the label, $s'w$, of the corresponding child of g'', which would have been assigned to it according to (a).

We can see that the suffix is constant within inserted parts and the prefix is constant within subtrees $G^{**}(g,s)$. Evidently, the suffix shows the path in G^{**}

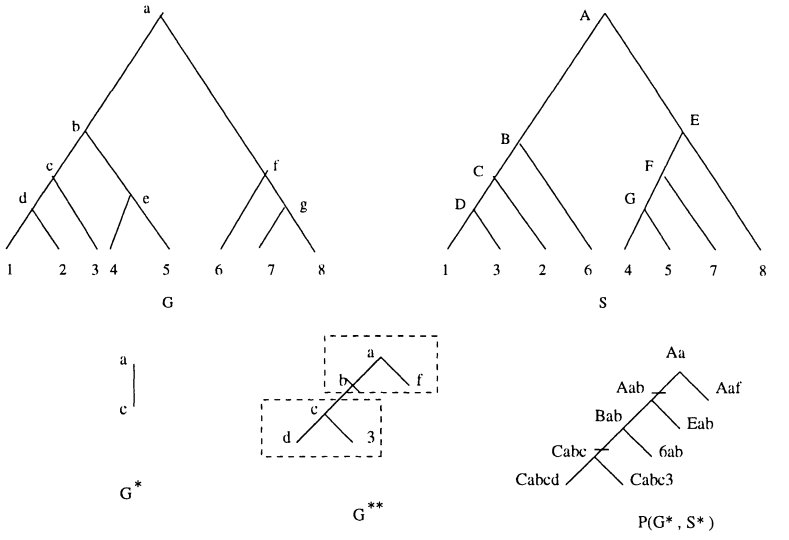


FIGURE 11. The plait frame, $P(G^*, S^*)$, for a gene tree, G, and species tree, S as an extended and relabeled structure of gene subtrees, G^* and G^{**}.

leading from the root to the duplicate copy of the gene at the correspondingly labeled species node: any time when a duplication occurs (at a gene node g), the duplicate copies are distinguished by adding to them "marking signs" corresponding to its children, cg and $\bar{c}g$. Obviously, any suffix assigned in the plait frame, $w = g_1...g_p$, satisfies inclusions $g_p \subset ... \subset g_1$ along the corresponding path in G^{**} where $g_1, ..., g_p$ are considered as clusters in G.

In the gene and species trees shown in Fig. 11, tree G^*, tree G^{**} and the labeled plait frame $P(G^*, S^*)$ are present. The inserted part (between A and C) in S is shown in tree $P(G^*, S^*)$ with cuts.

The plait frame sketches the duplication history of gene G according to species tree S. To show a complete history involving the list of all current species, I, under observation, the *plait tree*, $H(G, S)$, can be defined as obtained from S by substituting subtree $S(s^*)$ (where s^* is the root of S^*) by the plait frame with each of its leaves sw replaced by the subtree $S(s)$ along with all its nodes s' relabeled as $s'w$ by adding the suffix w of the plait tree leaf (see Fig. 12). An important feature in this representation is the labels of the leaves of $H(G, S)$ presenting joint duplication history of gene G in the species set I under investigation. Any leaf labeled by $ig_1...g_p$ relates to a copy of gene G in species i. If $i \notin g_p$, the leaf copy is extinct or just missing in the data. If $i \in g_p$, the leaf $ig_1...g_p$ corresponds to the observed occurrence of the gene in species $i \in I$. This is why we mark leaves $ig_1...g_p$ with $i \in g_p$ as active while the others are marked as non-active. The non-active copies in the plait tree in Fig. 12 are marked by the asterisk, "*".

It can be easily proven that the plait tree, in fact, coincides with the reconciled tree as does the tree in Fig. 12 with the tree in Fig. 7 (up to minor labeling differences). To do that, we need to translate the format of encoding plait tree nodes, $sw \in S \times W$, into the format of labeling nodes of the reconciled tree. In the

latter case, the suffix must be a subset of only copy labels. Such a subset, $c(w)$, is obtained, for any $sw \in H(G,S)$, by picking from w those of its symbols that relate only to children of the duplications occurring along the path leading to sw. For example, for the node $Cabcd$ in $H(G,S)$ in Fig. 12, the copy set is $\{b,d\}$, because a and c are themselves duplications.

STATEMENT 10. *The plait tree $H(G,S)$, with its node label suffix words, w, relabeled as subsets $c(w)$, is the labeled reconciled tree $R(G,S)$.*

Proof: The rule for updating the natural partial order in the algorithm for constructing the labeled reconciled tree follows the structure of plait frame $P(G^*,S^*)$. The suffixes are added only from the children cg and $\bar{c}g$, not from the duplicate node g itself. These additions correspond to gene copies. □

4.3. Losses and *-Losses. To analyze the structure of the set of *-losses, we use the subtrees $G^{**}(g,s)$ of the subtree G^{**} defined in section 4.1. For any leaf, g_k, $k = 1,...,m$, of $G^{**}(g,s)$, the set of non-active copies in the correspondingly relabeled subtree $S(s)$ is $s - g_k$. This is obvious if g_k is a leaf of G^{**}. However, this also holds if g_k is not a leaf of G^{**} because it is substituted by that piece of S which contains $s - g_k$, because further duplications (and copying process) are within g_k as its set-inclusion descent. Thus corresponding *-losses are maximal nodes $s' \in S$ satisfying $s' \subseteq s - g_k$.

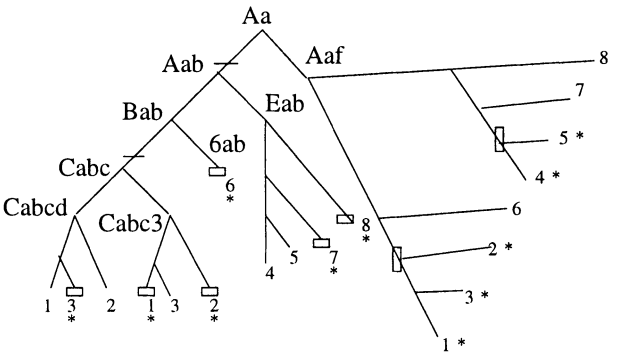


FIGURE 12. The plait tree $H(G,S)$, for the gene and species trees in Fig.1. The label suffixes are omitted below the plait frame nodes, for notational simplicity.

In the example of Fig. 12, there are two subtrees corresponding to children b and f of duplication a with their roots Aaf and Aab, respectively. Since $A = I$ in this case, corresponding sets of non-active copies are $A - f = \{1,2,3,4,5\}$ and $A - b = \{6,7,8\}$, respectively. The *-losses are shown by boxes as the maximal species clusters of non-active copies.

The following two statements show that the concepts of loss and *-loss are equivalent.

STATEMENT 11. *The set of losses corresponding to all the duplication nodes of any subtree $G^{**}(g,s)$ of S^{**}, coincides with the set of *-losses in the corresponding part of $H(G,S)$ (with the number of occurrences of the same loss taken into account).*

Proof: Let us recall that a g'-loss, $s' \in S$, is defined by the condition that ($s' \cap g' \subseteq cg'$ or $s' \cap g' \subseteq \bar{c}g'$) and $s' \cup s''$ overlaps both cg' and $\bar{c}g'$, where s'' is the sibling of s' in S. Say $s' \cap g' \subseteq cg'$ and $s'' \cap \bar{c}g' \neq \emptyset$; thus there exists $i_0 \in s'' \cap \bar{c}g'$. By construction of G^{**}, there exists a leaf $g'' \in G^{**}$, $g'' \subset g'$, such that $i_0 \in g''$. Therefore, there exists a leaf $i_0 w g'' \in H(G,S)$ such that $i_0 \in g''$ and w contains cg' as its letter. This implies that in the corresponding subtree of S carrying label suffix wg'' in $H(G,S)$, the species cluster s' contains only non-active copies but $s' \cup s''$ does not, which means that s' is a *-cluster.

To prove that there are no other *-clusters, let us make the following calculation. Let $g_1, ..., g_m$ be the leaves of $G^{**}(g,s)$. As we have seen above, the *-losses are maximal species clusters in $s-g_1$,..., $s-g_m$ which implies that they cover $m-1$ times every $i \in s$. On the other hand, according to Statement 1, the losses corresponding to each of the $m-1$ duplication nodes in $G^{**}(g,s)$ form a partition of s so that these partitions cover s also $m-1$ times, which implies that there cannot exist any *-loss not being a loss. □

Summing up all the losses (and *-losses) in all $G^{**}(g,s)$, we have the following corollary proven.

STATEMENT 12. *The sets and numbers of all losses and *-losses coincide.*

5. Conclusion

Two concepts of duplication, apparently different though based on similar ideas, have been analyzed here. It is proven that the patterns of duplications and their histories emerging in two approaches to modeling duplications are equivalent in the special case that each species contains exactly one gene of a duplicated gene family. One of the approaches deals with copying duplications presented in the reconciled tree, the other with annotating duplications.

In the reconciled duplication history tree all duplications are inferred in such a way that the tree shows how the evolutionary process might have occurred. In particular, the nesting of duplications is easily seen. In contrast, the annotating duplications may be inferred in any order, which makes Mirkin-Muchnik-Smith's annotating construction not so graphical. Moreover, as noted by Page [**17**] the reconciling techniques can be easily extended to the case when several gene copies can be present for the same species (see also section 2.4 in this paper). The problem of extension of the annotating techniques to the case of multiple gene copies has, to the authors' knowledge, never been explored.

However, some good features can be found in the latter approach, too: (i) it allows separating those non-active leaves that definitely are based on a lack of data rather than on real loss - this is the essence of the "gapped loss" concept (though some of the other non-actives also can be due to a lack of data); (ii) several gene trees can be mapped with the annotating duplications onto the same species tree, thus admitting multifaceted evolutionary interpretation, which hardly can be done with the reconciling representation.

Most amazingly, these two biologically meaningful constructions appear to be highly connected with a purely combinatorial approach based on the least-common-ancestor mapping. It is proven that the combinatorial cost function gives exactly the number (though not location) of losses/*-losses. Finally, we have shown that counting the intermediate nodes is equivalent to counting the duplications for which the collateral children are losses.

The least-common-ancestor tree mapping and counting the intermediate nodes (simultaneously, as Eulenstein [**4**] does, or subsequently, as suggested in Zhang [**19**]) can be done in a linear time (over the size of the trees), which makes tree-mapping a welcome instrument in comparing and reconciling gene/species evolutionary trees. The question of how computationally expensive it is to enumerate all the losses (with regard to corresponding duplications) remains open: the annotating and reconciling duplication constructions here requires cubic time (over the size of the trees).

The concepts considered here can be further investigated. For instance, our concept of the labeled reconciled tree involves given species and gene trees. However, characteristics of the reconciled or plait tree should be found in general terms, with no particular species/gene tree given. The problems of revealing corresponding gene and species structures from an abstract reconciled tree have yet to be posed and solved.

Also, it is interesting to investigate what kind of interrelation exists between the duplication/loss measures analyzed in this paper and those tree-difference measures developed in the literature earlier (see, for example, [**1**], [**2**] and [**11**]).

6. Acknowledgements

The authors are indebted to I. Muchnik, R. Page, T. Smith and L. Zhang for presenting their results while unpublished. The authors thank G. Estabrook for his numerous revising suggestions.

References

[1] H. Bobisud and L. Bobisud, A metric for classifications, Taxon. 21 (1972) 607 – 613.
[2] W.H.E. Day, Optimal algorithms for comparing trees with labeled leaves, Journal of Classification. 2 (1985) 7 – 28.
[3] G. Estabrook and F. McMorris, When is one estimate of evolutionary relationships a refinement of another?, J. Math. Biology. 10 (1980) 367 – 373.
[4] O. Eulenstein, A linear time algorithm for tree mapping. "Arbeitspapiere der GMD" (1997) No. 1046, Germany.
[5] O. Eulenstein, B. Mirkin and M. Vingron, Duplication-based measures of difference between gene and species trees. (1997) submitted.
[6] O. Eulenstein, B. Mirkin and M. Vingron, Modeling joint history of duplications in evolutionary trees. (1997) submitted.
[7] O. Eulenstein and M. Vingron, On the equivalence of two tree mapping measures, "Arbeitspapiere der GMD" (1995) No. 936, Germany.
[8] J. Felsenstein, Phylogenies from molecular sequences: Inference and reliability, Annu. Rev. Genet. . 22 (1988) 521 – 565.
[9] W. Fitch and E. Margoliash, Construction of phylogenetic trees, Science. 155 (1967) 279 – 284.
[10] M. Goodman, J. Czelusniak, G. Moore, A. Romero-Herrera, and G. Matsuda, Fitting the gene lineage into its species lineage: A parsimony strategy illustrated by cladograms constructed from globin sequences, Syst.Zool. 28 (1979) 132 – 168.
[11] A. Gordon, Hierarchical classification, in P. Arabie, L. Hubert, and G. De Soete (Eds.) Classification and Clustering (World Scientific, Singapore, 1996).
[12] R. Guigó, I. Muchnik, and T. F. Smith, Reconstruction of ancient molecular phylogeny. Molecular Phylogenetics and Evolution. 6 (1996) 189 – 213.
[13] W. Li and D. Graur. Fundamentals of Molecular Evolution (Sinauer Associates, Inc., Sunderland, Massachusetts, 1991).
[14] B. Mirkin, I. Muchnik, and T. F. Smith, A Biologically Consistent Model for Comparing Molecular Phylogenies, Journal of Computational Biology. 2 (1995) 493 – 507.
[15] M. Nei. Molecular Evolution Genetics (Columbia University Press, New York, 1987).

[16] G. Nelson and N. Platnick. Systematics and Biogeography: Cladistics and Vicariance (Columbia University Press, New York, 1981).
[17] R. D. Page, Maps between trees and cladistic analysis of historical associations among genes, organisms, and areas, Systematic Biology. 43 (1994) 58 – 77.
[18] J. Maynard Smith. The Theory of Evolution (Penguin Books Ltd, Harmondsworth, Middleessex, England, 1958).
[19] L. Zhang, A Proof of a Mirkin-Muchnik-Smith Conjecture for Comparing Molecular Phylogenies, Journal of Comp. Biology. (1997) to appear.

UNIVERSITY OF BONN, DEPT. OF COMPUTER SCIENCE, RESEARCH GROUP OF PROF. LENGAUER, RÖMERSTR. 164, D-53117 BONN, GERMANY.
E-mail address: Oliver.Eulenstein@gmd.de

DIMACS, RUTGERS UNIVERSITY, P.O.BOX 1179, PISCATAWAY NJ 08855, USA AND DKFZ, ABTEILUNG THEORETISCHE BIOINFORMATIK, HEIDELBERG, GERMANY.
E-mail address: mirkin@dimacs.rutgers.edu

DKFZ, ABTEILUNG THEORETISCHE BIOINFORMATIK, INF 280, D-69120 HEIDELBERG, GERMANY.
E-mail address: m.vingron@dkfz-heidelberg.de

This page intentionally left blank

https://doi.org/10.1090/dimacs/037/06

DIMACS Series in Discrete Mathematics
and Theoretical Computer Science
Volume **37**, 1997

Fitting Models of Intron Evolution to Aldehyde Dehydrogenase Data

Andrey Rzhetsky, Francisco José Ayala, Lily C. Hsu, Cheng Chang, and Akira Yoshida

ABSTRACT. Whether or not nuclear introns predate the divergence of bacteria and eukaryotes is the central argument between the proponents of the "introns-early" and "introns-late" theories. In this study we compared the "goodness-of-fit" of each theory using a probabilistic model of exon/intron evolution and new genomic sequences of non-allelic genes encoding human aldehyde dehydrogenases (ALDH). Using a reconstructed phylogenetic tree of ALDH genes we computed the likelihoods of obtaining the present-day ALDH sequences under the assumptions of each competing theory. Although *on the grounds of their own assumptions* each theory fit the ALDH data significantly better than its rival, the model corresponding to the "introns-early" theory required extensive "intron slippage," and the estimated slippage rates were too high to be consistent with previously reported correlations between the boundaries of ancient protein modules and the ends of ancient exons. Arguing that the molecular mechanisms proposed for explaining intron slippage are incapable of providing such high slippage rates and are incompatible with the observed intron distribution in higher eukaryotes, we concluded that the ALDH data support the "introns-late" theory.

1. Introduction

The "introns-early" theory suggests that the "genes in pieces" structure of eukaryotic genes emerged long before the Eubacteria, Archaebacteria, and Eukaryota diverged as separate groups (1, 2, 3). According to this theory, (*i*) the present-day exon/intron structures originated through the aggregation of short primordial mini-genes (15-20 amino acids) which were critically important for generating protein diversity through "exon shuffling," (*ii*) the apparent absence of spliceosomal introns in bacterial and organelle genomes resulted from their secondary loss, and (*iii*) the nuclear splicing machinery is as ancient as are the nuclear introns themselves. Furthermore, the theory presumes that introns can be easily lost and an "intron slippage" mechanism exists which

---

1991 *Mathematics Subject Classification.* Primary 92B05, 60G35, 65F15, 65L99.

This study was supported by grants from the National Science Foundation (#DEB 9520832) and the National Institute of Health (#GM20293-26) to Masatoshi Nei and and US Public Health Service Grant No. HL-29515 to Akira Yoshida. The authors are grateful to Masatoshi Nei, Jeffrey D. Palmer, Tanya Sitnikova, Yasuo Ina, Koichiro Tamura, Austin Hughes, Sergey N. Rodin, and Blair Hedges for numerous comments on the earlier versions of this paper.

can displace introns for short distances (1-12 nucleotides; see 4) while leaving the coding sequence intact.

The alternative "introns-late" theory (5, 6, 7) states that (*i*) split genes appeared by random intron insertion into primordial continuous protein-coding regions, (*ii*) the genes of cellular organelles and those of bacteria never had spliceosomal introns, and (*iii*) the spliceosomal machinery emerged through coevolution of group II self-splicing introns with eukaryotic proteins (8, 6, 9). Although the "introns-late" theory denounces the shuffling of primordial exons, it does not deny neither the possibility of recent exon shuffling within eukaryotic lineages nor early protein evolution by fusion, duplication, and permutation of primordial protein modules. However, the "introns-late" theory does not permit intron slippage and thus regards all introns occupying different sites within related proteins as non-homologous.

The following lines of argument have been used to either support or reject the two theories. (*i*) A few introns were found in homologous positions in genes duplicated *before* the separation of eukaryotes and bacteria (supporting "introns-early") (10), although the distribution of the vast majority of introns in such genes seems to be better explained by intron insertion (9). (*ii*) "Introns-early" supporters correctly predicted the position of a new intron in a gene of mosquito *Culex tarsalis* (11), although this was later argued to be a lucky coincidence (12, 13, 14). (*iii*) "Introns-early" supporters have claimed that exon/intron boundaries statistically correlate with the ends of units of protein three-dimensional structure (ancient "modules," *e.g.,* see 15), although this conclusion was also vigorously challenged (14, 16). (*iv*) Multigene analyses of the distribution of intron phase indicated a significant excess of exons and exon groups with the same intron phase at both ends (which was presented as evidence for the "introns-early" theory; 17, 18), but this could have resulted from recent exon shuffling events, and is thus compatible with both theories (13). (*v*) Parsimonious reconstructions of the evolution of the exon/intron structure in eukaryotes supported the "introns-late" view (9, 14) but the possibility of intron slippage was completely discarded in these analyses.

We present here a new method aimed at (*i*) qualitatively analyzing new data under the respective assumptions of the two competing theories, (*ii*) scrutinizing the internal consistency of the results of each analysis, and (*iii*) evaluating factual support for the assumptions underlying each theory. We illustrate the application of this new method with an analysis of aldehyde dehydrogenase (ALDH) genes.

2. Human ALDH genes

2.1. Human ALDH genes are ancient. Aldehyde dehydrogenases are enzymes catalyzing the conversion of biogenic and foodstuff aldehydes into acid metabolites (19, 20, 21). Humans have at least ten homologous ALDH genes that apparently emerged from a series of duplications of a single ancestral gene (22, 23, 24, 25, 26, 27, 28, 29), and which have surprisingly diverse exon/intron structures (fig. 1, 2). Although all known human ALDH's are nuclearly encoded, at least three of them (ALDH2, ALDH5, and methylmalonate-semialdehyde dehydrogenase, MMSDH) have leader peptides and are transported to the mitochondria after synthesis.

A neighbor-joining tree of *ALDH*-like sequences from several eukaryotic and prokaryotic species yielded four well-defined clusters of eukaryotic genes (fig. 3).

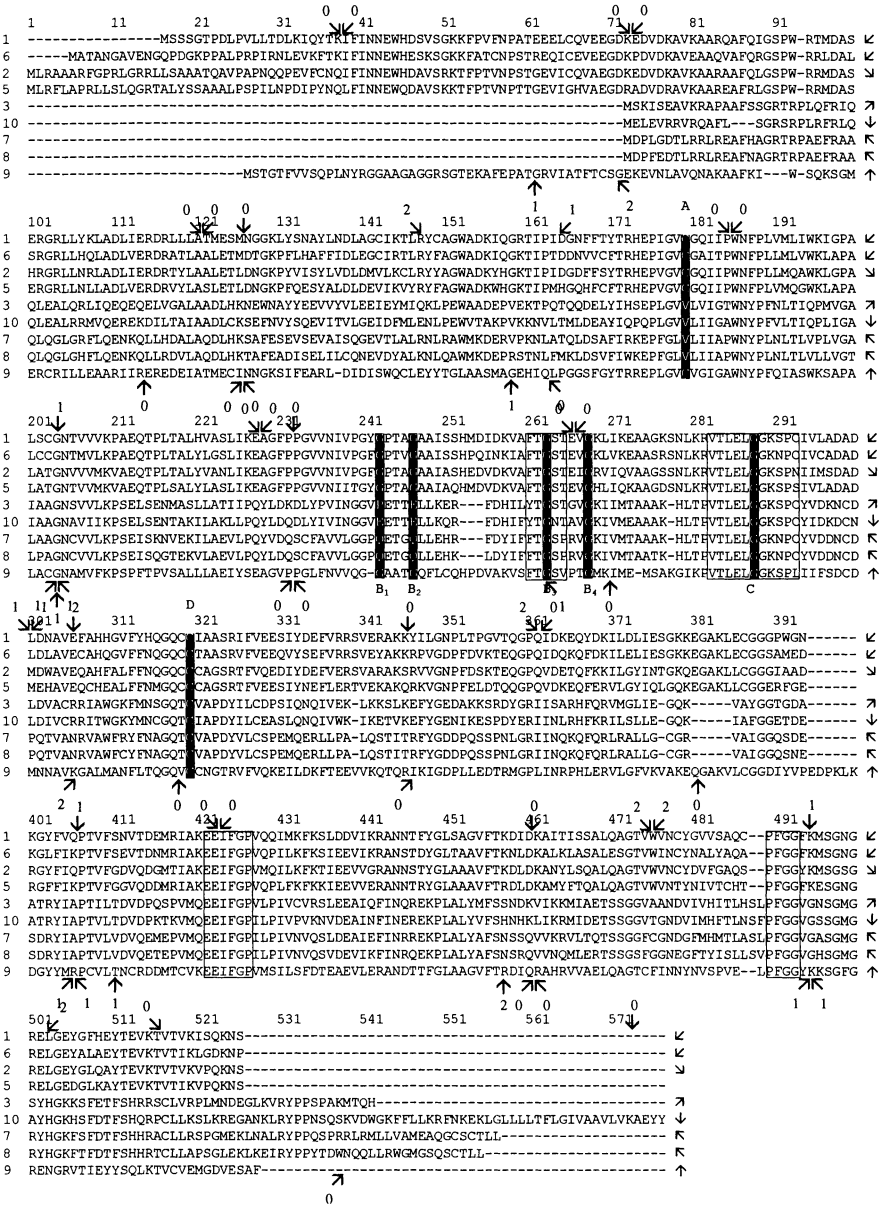


Fig. 1. Positions of introns within eight human genes superimposed with alignment of corresponding protein sequences. Intron positions (arrowheads) for ALDH1/2/5/6/10 and ALDH3/7/8/9 groups are shown above and below the alignment, respectively. The number shown next to each arrowhead (0, 1, or 2) indicates the "phase" of corresponding intron with respect to the reading frame (0 - exon/intron boundaries are between codons, 1 - after the first codon position, and 2 - after the second nucleotide in the codon). Open boxes emphasize conservative sites; closed boxes show amino acid sites with established function. Although we were able to find alternative plausible alignments, all of them predicted the same relative arrangement of introns.

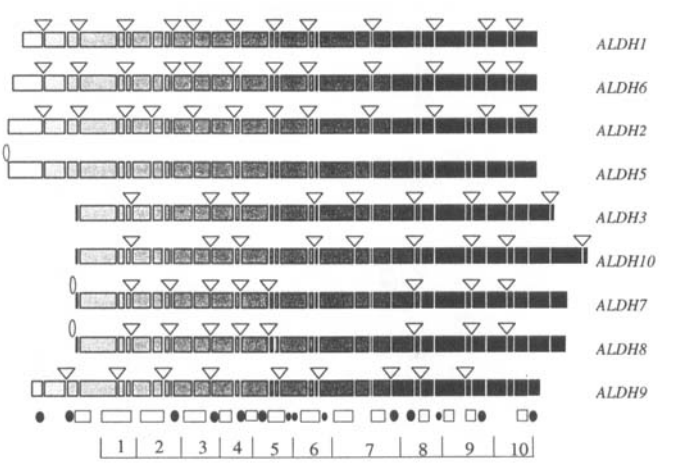


Fig. 2. Exon/intron structures of human ALDH genes mapped to the alignment of their amino acid sequences. The sequences themselves are shown as shaded rectangles where the shading intensity increases in direction from N- to C-terminus of the protein; deletions and insertions are not shown. Each discontinuity in rectangles corresponds to an exon/intron boundary observed in at least one of the genes; triangles and ellipses indicate only those introns that are actually found in the corresponding gene. The figure also shows the predicted secondary structure that is assumed to be similar for all compared proteins: the hatched boxes indicate α-helices and the filled circles correspond to β-strands. The secondary structure was predicted with a neural network algorithm implemented in program PHD (30, 31). The bottom of the figure shows an artificial segmentation of the protein into ten domains which was used in computation of the likelihood values.

Although the average substitution rate in the group IV cluster (*ALDH3/7/8/10*) was twice as large as the rate in the group I cluster (*ALDH1/2/5/6*), each of the two groups *separately* conformed to a "molecular clock" (see figs. 3 and 5 A) allowing an estimation of the divergence times between genes (see 32, 33). These estimates (fig. 5 A) indicated that duplications in group I were likely to have occurred much earlier than the duplications in group IV. In our reconstruction, diversification within group I happened during the Neoproterozoic period (34), while duplications in group IV seemed to arise much later, in the Phanerozoic period, when diverse vertebrate and invertebrate animals were already abundant. The latest two duplications in group IV probably took place near 212 and 87 million years ago, respectively (fig. 5 A), dates which roughly correspond to the appearance and then subsequent radiation of mammals. Finally, the existence of at

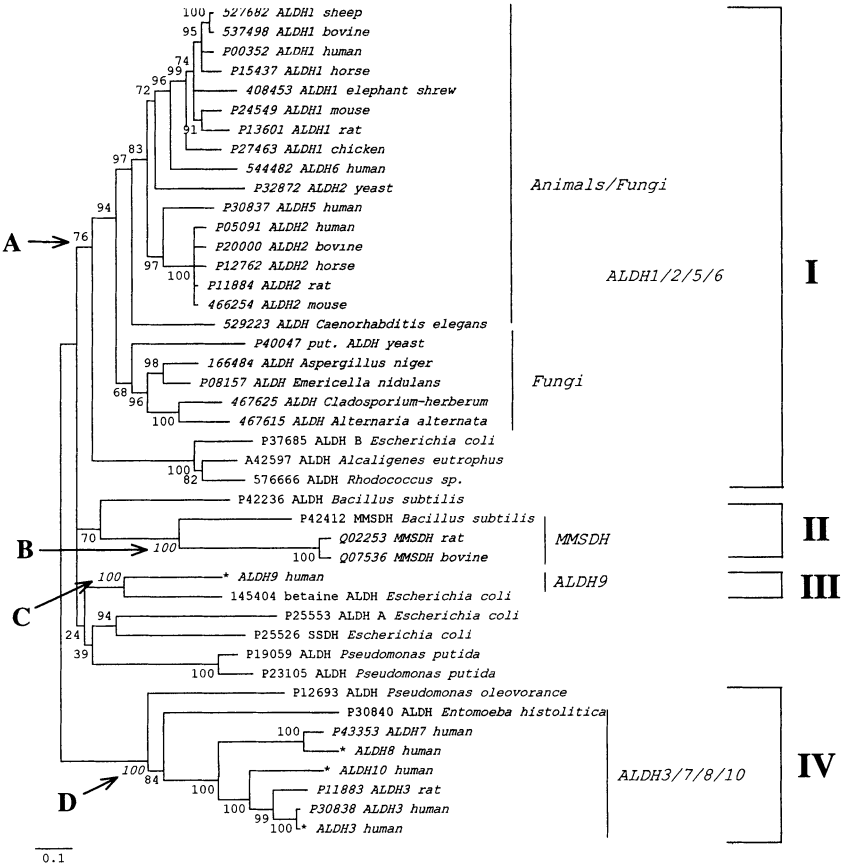


Fig. 3. A neighbor-joining tree (35) computed and visualized with MEGA (36) from 43 ALDH-like protein sequences using the Poisson correction (37, for ALDH data virtually all currently available corrections for multiple hits give essentially the same tree) for multiple hits; the branch lengths are given in terms of the number of amino acid substitutions per site. All sites with deletions or insertions were excluded from the analysis. The shaded areas indicate sequences from eukaryotic organisms. We deliberately excluded plant ALDHs from the analysis to facilitate interpretation of the resulting phylogeny. Bootstrap p-values are shown next to the corresponding interior branches; the interior branches which were supported with 20% or less out of 500 bootstrap replications (39) were set to zero. Description of each protein sequence includes either SwissProt or GenBank accession number (asterisks indicate new sequences) and protein and species names. **A**, **B**, **C**, and **D** indicate the interior branches defining four stable clusters of proteins from both eukaryotes and eubacteria. The tree may indicate that at least four ALDH-like genes (*ALDH1/2/5/6*-like, *ALDH3/7/8/10*-like, *ALDH9*-like, and *MMSDH*-like genes) pre-existed the divergence of eukaryotes and eubacteria. SSDH stands for succinate-semilaldehyde dehydrogenase.

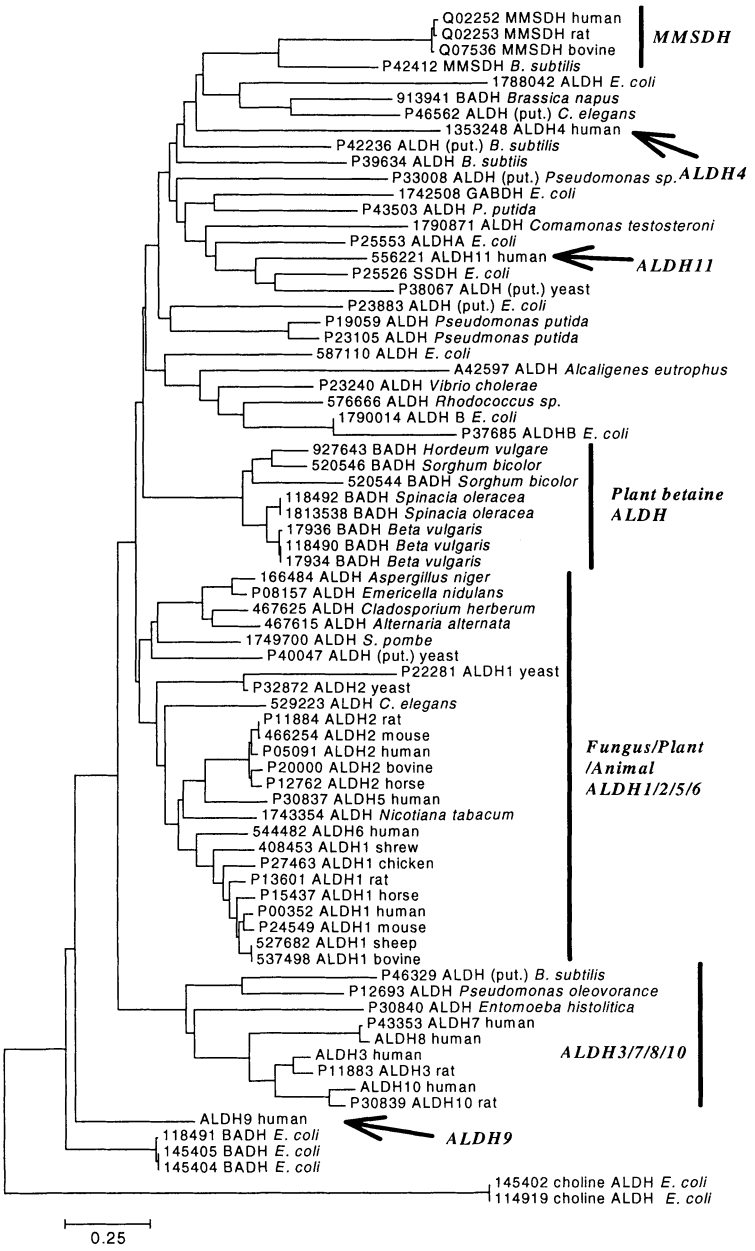


Fig. 4. A neighbor-joining tree computed with MEGA (36) and visualized with TREEVIEW program by K. Tamura from expanded sample of ALDH-like protein sequences including plant sequences. As in the previous figure, the Poisson correction (37) for multiple hits was applied; the branch lengths are given in terms of the number of amino acid substitutions per site. All sites with deletions or insertions were excluded from the analysis. Although this tree is considerably less stable than the one on the previous figure when tested with bootstrap (data not shown), it clearly shows that there is either fewer different non-allelic ALDH genes in plants than in animals, or many of the plant ALDH genes are not discovered yet.

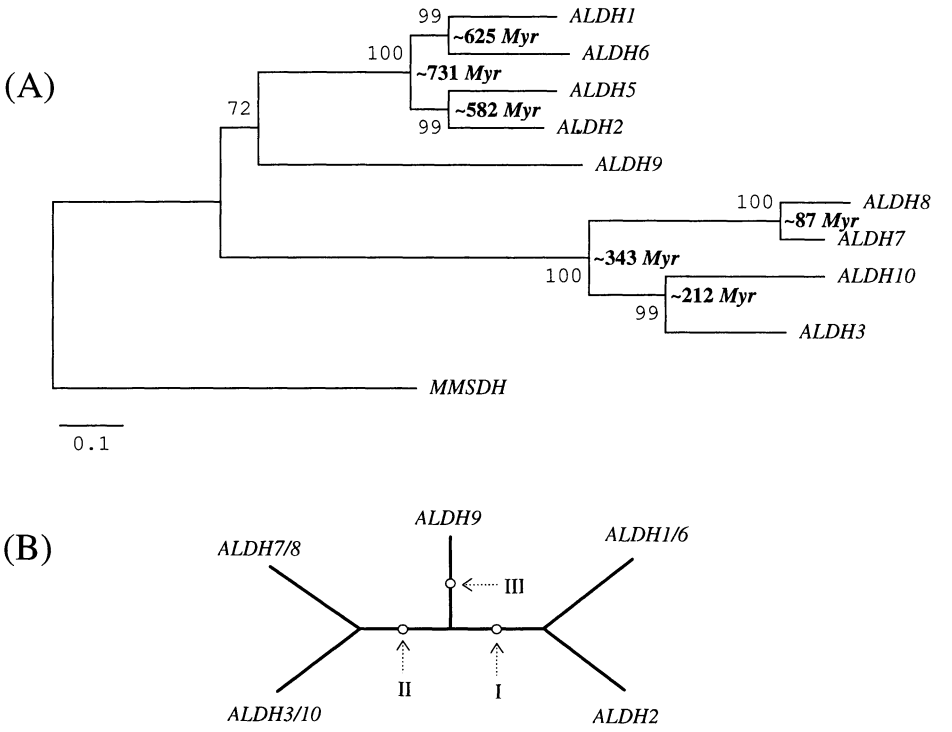


Fig. 5. (A) A neighbor-joining tree (35) computed and visualized with MEGA (36) from eleven human protein sequences using the Poisson correction for multiple hits (37). The per cent of bootstrap (39) resamplings (out of 500) supporting each sequence partition is shown next to the corresponding interior branch; the times of divergence between human genes were estimated with the "linearized tree" algorithm (32). In this estimation we used known ALDH protein sequences from Rodents, Primates, and Artyodactyls and the divergence time 104 *Myr* (33) for Rodentia/Artiodactyla bifurcation. (B) The unrooted tree topology that was used in the maximum likelihood analysis of exon/intron organization of ALDH genes. There are three pairs of ALDH genes which have identical exon/intron patterns within each pair: *ALDH1* and *ALDH6*, *ALDH3* and *ALDH10*, and *ALDH7* and *ALDH8*. The unrooted tree topology is the same as the neighbor-joining trees in figs. 3 A and 4. The arrows show three alternative positions of the tree root (in the maximum likelihood computation we refer to these rooted trees as tree I, tree II, and tree III, see Table 1).

least four clusters of ALDH genes where bacterial and eukaryotic genes are grouped together (see figs. 3, 4) suggested that the divergence times of the four clusters are greater than 2110 *Myr*, the estimated age of the oldest known extinct eukaryote, *Crypania spiralis* (38).

Our phylogenetic reconstruction thus indicated that the "progenote," the common ancestor of eukaryotes and bacteria, was likely to have at least four distinct ALDH genes, since animal and eubacterial genes were grouped together with high bootstrap (39) support. (At least some of the hypothetically ancient homologous ALDH genes are found in five kingdoms of living organisms, Bacteria, Protozoa, Plants, Fungi, and Animals, although plant ALDH genes seem to be studied less extensively than the animal genes, see fig. 4.) An alternative explanation of the same tree would require three or more "late" (after the eukaryotes/bacteria divergence) lateral gene transfers between animals and

bacteria. Either explanation should be compatible with the maximum likelihood analysis presented in the following section.

2.2. Intron evolution in ALDH genes: comparison of competing theories. We developed a model that was flexible enough to account for analyses under each rival theory. Our model incorporates the following assumptions. (*i*) There are three major types of elementary events causing changes in exon/intron patterns: intron insertion, intron deletion, and intron slippage, where an intron slippage is a hypothetical short-range "jump" of an intron within the same gene. (*ii*) The actual number of elementary events along a tree branch follows a Poisson distribution. (*iii*) The probability of each new evolutionary event *given a fixed exon/intron arrangement* does not depend on either the order or the number of past events in the evolutionary history of the gene. (*iv*) Rates of intron insertion, deletion and slippage are fixed along each branch of the tree, but can differ among branches. We also assumed that the correct *unrooted* tree topology for human ALDH genes is known and can be meaningfully rooted in three alternative ways (fig. 5 B).

Starting with the above assumptions, we applied a standard set of matrix manipulations (40) used in the theory of Markov chains for deriving transition probabilities between different exon/intron patterns. These probabilities were then used to compute the conditional probability ("the likelihood given data") of observing the present-day gene structures given a specified tree and a fixed set of the model parameter values (41). First, we defined instantaneous transition rate matrices corresponding to a first-order Markov chain description of intron evolution. The entries of each matrix were assigned rate parameters λ, μ, or ϕ whenever the corresponding pair of intron arrangements was separated by a *single* intron insertion, deletion, or slippage, respectively; the matrix entries were set to zero whenever the distance between corresponding intron arrangements exceeded one elementary event. The diagonal elements of each rate matrix were chosen to ensure that the sum of elements in each row is equal to zero. For example, for a hypothetical gene with only *two* sites potentially hosting introns, there are four possible intron/exon configurations: 00, 01, 10, and 11, where zero and one stand for intron absence and presence, respectively. Thus, the transition from configuration 00 to configuration 01 corresponds to an intron insertion; the transition from 01 to 00 indicates intron loss; a transition from 10 to 01 denotes an intron slippage. The resulting instantaneous transition rate matrix, $\mathbf{Q}$, is then written as follows

$$\mathbf{Q} = \begin{bmatrix} -2\lambda & \lambda & \lambda & 0 \\ \mu & -\lambda-\mu-\varphi & \varphi & \lambda \\ \mu & \varphi & -\lambda-\mu-\varphi & \lambda \\ 0 & \mu & \mu & -2\mu \end{bmatrix} \begin{matrix} 00 \\ 01 \\ 10 \\ 11 \end{matrix}$$
$$\begin{matrix} 00 & 01 & 10 & 11 \end{matrix}$$

where λ, μ, and ϕ stand for the instantaneous rates of intron insertion, deletion, and slippage, respectively. Second, the matrices of transition probabilities between exon/intron arrangements were computed numerically as matrix exponentials of the corresponding instantaneous transition rate matrices. This operation produces a matrix of transition probabilities between gene arrangement states during time t (expressed in terms of the expected number of events of each type), and is symbolically expressed as $e^{\mathbf{Q}t}$. Third, the likelihood value was calculated as described by J. Felsenstein (41), treating the number of ancestral introns at the "root" of the tree and the mean rates of intron

rearrangement along each tree branch as model parameters. All numerical computations were performed with the MATLAB® 4.0 package produced by Math Works Inc. (To make the required computations feasible we divided the ALDH genes into ten domains (see fig. 2) assuming that intron slippage was prohibited between domains. We defined the boundaries between these domains to minimize the number of the ancestral introns required to explain the present-day genes, as is commonly done in "introns-early" analyses. Without this segmentation the computation of likelihood functions would be effectively impossible because of the large number of intermediate sequence states at each node of the tree. Indeed, each sequence with n potentially intron-bearing sites can be observed in 2^n different binary states, where 0 stands for an intron absence, and 1 for an intron presence. This is a very large number even for a moderate n (*e.g.*, more than 10^9 for $n = 30$), and the likelihood values have to be computed by evaluating transition probabilities through each of 2^n states for each interior node of the tree. Fortunately, it was possible to compute an *approximate* likelihood value by assuming that intron slippages can move introns only *within* each of the ten domains shown in the figure. Only the present-day intron positions were used for the computation.) Finally, we used multidimensional simplex numerical optimization to find a set of parameter values maximizing the likelihood value. (We eliminated *ALDH5* from the analysis because this gene apparently resulted from a single processed mRNA reverse transcription event.)

With this model we were able to directly compare the fit of each alternative theory to the actual ALDH data. The fit of any two models to the data set can be objectively compared with the Akaike Information Criterion (AIC; ref. 42). The AIC value is computed for each rival model according to a simple formula, $AIC_i = 2\ N_i - 2 \log L_i$, where N_i is the number of parameters used in the ith model, and $\log L_i$ is the logarithm of the maximum likelihood value obtained under the model. The criterion is designed such that the models that fit the data *better* have *smaller* AIC values.

The results of each analysis were completely different for each set of assumptions. Comparison of AIC values (Table 1, scenarios A, B, and C) showed that the probability of generating the actual ALDH data under the "no-slippage" assumption and the "insertions only" model (= "introns-late") was almost 10^6 times as large as the analogous probability under the "deletions only" (= "introns-early") model. To our surprise, reanalysis of the same data allowing for "intron slippages" ("introns-early" assumption, see Table 1, D, E, and F) resulted in a complete reversal of the conclusion. That is, the model "deletions + slippages" (= "introns-early") became the best with a large advantage in AIC values.

Thus, the "intron slippage" assumption is critical for discriminating between the two theories. Below we scrutinize the consistency of the available experimental data with the parameter estimates obtained in our maximum likelihood analysis. We demonstrate that although the model with the smallest ("best") AIC value corresponds to the "introns-early" theory (see Table 1 D), the parameter estimates obtained under this model appear incompatible with both available experimental data and previous arguments in favor of the "introns-early" theory.

Table 1. Comparison of alternative scenarios of exon/intron evolution in the maximum likelihood analysis.

Scenario	N[a]	ln$L$[b]	AIC[c]	Ancestral intron number	max branch slip[f]	max branch ins[g]	max branch del[h]
A. Only deletion							
Tree I/II/III	8	-176.75	369.51	31[d]	0.	0.	1.68
B. Only insertion							
Tree I	8 + 1	-165.75	***349.50***	0[e]	0.	0.02	0.
Tree II/III		-167.16	***352.32***		0.	0.02	0.
C. Insertion + Deletion							
Tree I	8 + 8 + 1	-165.75	365.50	0[e]	0.	0.02	0.
Tree II/III		-167.16	368.32		0.	0.02	0.
D. Deletion + Slippage							
Tree I		-92.68	***219.35***		11.6	0.	0.23
Tree II	8 + 8 + 1	-92.67	***219.34***	10[e]	7.9	0.	0.23
Tree III		-92.69	***219.38***		11.7	0.	0.23
E. Insertion + Slippage							
Tree I		-165.320	364.64		0.	0.02	0.
Tree II	8 + 8 + 1	-163.882	361.76	0[e]	0.	0.02	0.
Tree III		-165.992	365.98		0.	0.02	0.
F. Deletion + Insertion + Slippage							
Tree I	8 + 8 + 8 + 1	-92.66	235.33	10[e]	19.7	0.	0.22
Tree II/III		-92.67	235.34		22.4	0.	0.22

Note - [a] the number of model parameters, [b] the natural logarithm of the likelihood value, [c] Akaike Information Criterion, [d] the number of ancestral introns was pre-set rather than estimated, [e] the estimated number of ancestral introns, [f], [g], and [h] the maximum likelihood estimates of the rates of intron slippage, intron insertion, and intron deletion, respectively, expressed per site per branch of the tree.

The results of our analyses turned out to be completely different depending on the assumptions used. Comparison of AIC values (Table 1, scenarios A, B, and C) showed that the probability of generating the actual ALDH data under "no-slippage" assumption and the "insertions only" model (= "introns-late") was almost 10^6 times as large as the analogous probability under the "deletions only" (= "introns-early") model. To our surprise, the re-analysis of the same data allowing for "intron slippages" ("introns-early" assumption, see Table 1, D, E, and F) resulted in complete reversal of conclusion. That is, the model "deletions + slippages" (= "introns-early") became "the best" with a large advantage in AIC values. Apparently, the "intron slippage" assumption is critical for discriminating between the two theories and it is important to scrutinize the consistency of the available experimental data with the parameter estimates obtained in our maximum likelihood analysis.

Below we demonstrate that although the model with the smallest ("best") AIC value corresponds to the "introns-early" theory (see Table 1 D), parameter estimates obtained

under this model appear incompatible with both available experimental data and previous arguments in favor of the "introns-early" theory.

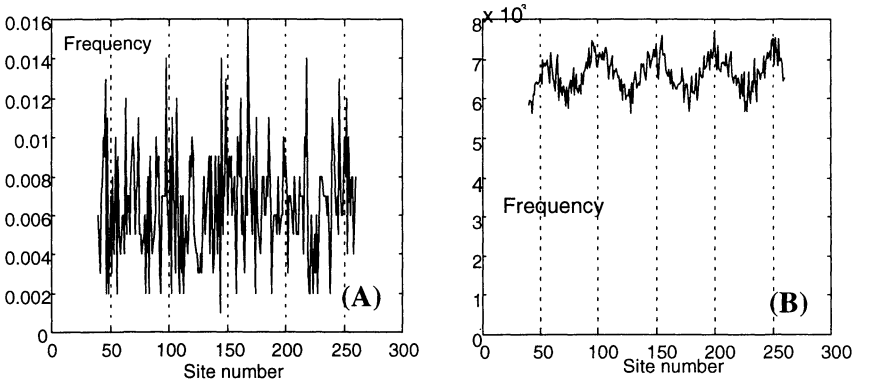


Fig. 6. (A) The non-randomness of intron distribution along gene is effectively undetectable when slippage rates are as high as was estimated in our maximum likelihood analysis and the number of independent present-day genes under analysis is not unrealistically large (<10,000). The figure shows a frequency distribution of introns which was computed through averaging *1000* "present-day" genes obtained in computer simulation. Each "present-day" gene independently "evolved" from a hypothetical ancestral gene with multiple "introns" separated by fifty-nucleotide "exons" (dashed lines indicate positions of the "ancestral" introns). Then, approximately two thirds of the "ancestral" introns were randomly deleted, the remaining introns were subjected to "slippages" at rate 9 slippages per site. Direction, 5' or 3', of each slippage event was chosen randomly; the length of each "leap" was sampled from a uniform distribution defined on interval [1, 12]. Note that in our simulation the "present-day" genes were assumed to evolve independently; the phylogenetic non-independence of actual present-day genes should additionally *increase* the variance of intron distribution. (B) To significantly prove non-randomness of intron distribution for the same model and parameter values one needs a very large sample of present day genes. This frequency distribution of intron positions was obtained by "averaging" over *100,000* (rather than 1000 in fig. 6 A) "present-day" genes generated as described above.

2.3. Mechanism of intron slippage and distribution of introns in human genes. At present there is no plausible known molecular mechanism to account for frequent intron slippage and that would be consistent with the actual patterns of intron distribution in eukaryotes. The simplest explanation for intron slippage is deletion of several nucleotides at the 3' end of one exon and insertion of the same number of nucleotides at the 5' end of the following exon. Since each of the two rearrangements by itself must be extremely deleterious (and putative intron slippages frequently leave the coding region undamaged) this mechanism appears to be inappropriate for explaining frequent slippage. Martinez *et al.*(43) suggested a more sophisticated mechanism based on the "single-intron-deletion" scenario of Fink (44). Fink's mechanism included the following steps: (*i*) a normal excision of an intron from pre-mRNA, (*ii*) reverse transcription of the modified pre-mRNA, and (*iii*) homologous recombination of the resulting cDNA with the original gene. Martinez *et al.* (43) hypothesized an additional event which may follow step (*i*): imprecise re-insertion of an excised intron back into the pre-mRNA (see ref. 45 for experimental evidence of reverse splicing). The advantage of the modified mechanism is that it accounts for a "clean" displacement of an intron within a coding region, although leaving the supposed twelve base-pair limit for intron slippage (4) unexplained. Since there is no direct evidence for reverse transcription of cellular RNAs

in eukaryotic cells, and since reverse transcription in retroviruses occurs only within the viral particle isolating cellular RNAs from the virus enzyme, the Fink-Martinez mechanism requires the presence of a *defective* retrovirus with a mutation in the packaging signal (44). (The simultaneous loss of several introns, as in the human *ALDH5* gene, can be explained by re-integration of the reverse-transcribed mRNA back into the genome (44, 46).)

Unless there exists strong selection preserving the number and/or spatial distribution of introns, evolution under the Fink-Martinez mechanism should result in very specific exon/intron structures (44): (a) Intron deletion should be more frequent than intron slippage, leading to a paucity of introns. (b) The retained introns should be concentrated near the 5' end of each gene because (*i*) reverse transcription begins at the 3' poly(A) tract of mRNA but rarely extends completely to the 5' end, and (*ii*) recombination between genes and cDNAs affects the ends of genes less frequently than the middle. (c) As a consequence of (a) and (b), intron slippages should be rarely observed at the ends of genes, especially at the 5' end. These predictions are in good accord with the exon/intron structures observed in yeast (44) but are clearly inconsistent with human ALDH genes: human genes have numerous introns which are *uniformly* distributed along the coding regions (see fig. 2), and to fit the "introns-early" theory hypothetical intron slippages have to be invoked at both ends of genes.

Unlike intron slippage, intron deletion can result from a one-step mutation event (rather than coincidence of two or more low-probability events), and, in the absence of counteracting selection, should be observed more frequently than intron slippage.

Finally, there are at least two hypothetical mechanisms explaining intron insertion. One is reverse splicing of an excised intron into a non-homologous pre-mRNA, followed by reverse transcription and homologous recombination (44). Another possible mechanism involves invasion of a group II intron (from organelles) into the nuclear genome, followed by a one-mutation transformation of the intron into a regular nucleosomal intron (6, 47, 48, 49, 50): only a single nucleotide substitution is required to convert "(U/C)A ... GU" dinucleotides flanking group II introns into canonical "GA ... GT" dinucleotides flanking nuclear introns, and it was recently discovered (51) that group II introns from yeast mitochondria can integrate *directly* into double-stranded genomic DNA . Therefore, the integration of group II introns into the genome is a one-step event where all molecular machinery is provided by the intron itself.

Thus, according to plausible evolutionary scenarios and the experimental evidence available today, intron slippage should be considerably less likely than intron deletion. In contrast, our maximum likelihood analysis under the "introns-early" assumptions (see Table 1 D, E, and F) suggested that to explain real data under this theory intron slippage has to be two orders of magnitude more frequent than intron deletion.

To demonstrate that the estimated rates if intron slippage contradict support for the "introns-early" theory based on a putative correlation between the ends of ancestral protein "modules" and the boundaries of proto-exons (*e.g.*, see 52), we performed a computer simulation built on the assumptions of the "introns-early" theory. This simulation (see fig. 6 A, B) demonstrated that the reported correlation cannot be detected from any *reasonable* sample of present-day genes (say, < 10,000) if intron slippage rates were as high as estimated in our analysis (see fig. 6 A, B). In our simulation "present-day" genes independently "evolved" from a hypothetical ancestral gene with multiple introns separated by fifty-nucleotide exons (dashed lines indicate positions of the "ancestral" introns). Approximately two thirds of the "ancestral" introns were then randomly deleted,

and the remaining introns were subjected to slippage at a rate of 9 slippages per site. The direction of each slippage event (either 5' or 3') was chosen randomly; the length of each "leap" was sampled from a uniform distribution defined by the interval [1, 12]. Figure 6 shows the resulting distribution of introns from a sample of 1000 genes (fig. 6 A) and 100,000 genes (fig. 6 B). (In our simulation the "present-day" genes were assumed to evolve independently; the phylogenetic non-independence of actual present-day genes should *increase* the variance of intron distribution.)

Assuming that introns were inserted into coding sequences relatively recently, how can one explain the non-randomness of intron distribution? Recent experimental data (*e.g.*, 53) indicate that nuclear DNA of eukaryotes is non-uniformly protected by proteins maintaining chromosome structure. For example, it was shown that during transcription of the *Dam* gene in yeast, each nucleosome associated with *Dam* selectively shielded approximately eighty base pairs of yeast DNA while allowing methylation enzymes to freely access DNA in internucleosome "linkers" (53). Therefore, we hypothesize that a non-uniform distribution of introns in eukaryotic genes could have been caused by preferential intron insertion into stretches of DNA that were temporary liberated from nucleosome protection.

3. Conclusion

The "intron slippage" assumption is the cornerstone of many lines of defense of the "introns-early" theory, yet, according to our analysis of ALDH genes it is precisely this assumption that leads to an internal contradiction between the arguments supporting the theory: first, contrary to expectation, the estimated intron slippage rates are much higher than the estimated intron deletion rates; second, high intron slippage rates question the reported correlation between the boundaries of the "ancient protein modules" and the ends of "proto-exons" (15). Indeed, if intron slippages are allowed, *each* putative ancestral intron had to move *at least once* to arrive at the present-day exon/intron arrangement in human ALDH genes. This is because *all* intron positions between groups *ALDH1/2/6* and *ALDH3/7/8/10*, and *ALDH3/7/8/10* and *ALDH9* are different (see fig. 2) and only *one* out of nine intron positions is conserved between *ALDH9* and *ALDH3/7/10*. Therefore, it is hardly surprising that the rates of intron slippage estimated in our maximum likelihood analysis are very high (Table 1).

In summary, the assumption of frequent intron slippage leads to inconsistencies with both the available body of experimental evidence and the data analyses provided by proponents of the "introns-early" theory; without this assumption the human ALDH data support the "introns-late" theory. The methods illustrated in this article can be readily applied to other data sets to test the generality of the conclusions drawn from the ALDH data.

4. References

1. Darnell, J.E., & Doolittle, W.F., *Speculations on the early course of evolution*, Proc. Natl. Acad. Sci. USA **83** (1986), 1271-1275.
2. Gilbert, W., Marchionni, M., & McKnight, G., *On the antiquity of introns*, Cell **46** (1986), 151-154.
3. Dorit, R.J., Schorndach, L, & Gilbert, W., *How big is the universe of exons?* Science **250** (1990), 1377-1382.
4. Cerff, R. *Tracing biological evolution in protein and gene structures,* eds. Gō , M., & Schimmel, P. Elsevier, New York, 1995, pp. 205-227.
5. Rogers, J.H. *The role of introns in evolution.* FEBS Lett. **268** (1990), 339-343.
6. Cavallier-Smith, T. *Intron phylogeny: a new hypothesis*, Trends Genet. **7** (1991), 145-148.
7. Patthy, L., *Exons -- original building blocks of proteins?* Bioessays **13** (1991), 187-192.
8. Sharp, P.A., *"Five easy pieces,"* Science **254** (1991), 663.

9. Palmer, J.D., & Logsdon, J.M., Jr., *The recent origins of introns*, Curr. Opin. Genet. Dev. **1** (1991), 470-477.
10. Kersanach, R., Brinkmann, H., Liaud, M.-F., Zhang, D.-X., Martin, W., & Cerff, R., *Five identical intron positions in ancient duplicated genes of eubacterial origin.* Nature (London) **367** (1994), 387-389.
11. Tittiger, C., Whyard, S., & Walker, V.K., *A novel intron site in the triosephosphate isomerase gene from the mosquito* Culex tarsalis, Nature (London) **361** (1993), 470-472.
12. Kwiatowski, J., Krawczyk, M., Kornacki, M., Bailey, K., & Ayala, F., *Evidence against the exon theory of genes derived from the triose-phosphate isomerase gene.* Proc. Natl. Acad. Sci. USA **92** (1995), 8503-8506.
13. Hurst, L.D., & McVean, G.T., *A difficult phase for introns-early*, Curr. Biol. **6** (1996), 533-536.
14. Logsdon, J.M., Jr., Tyshenko, M.G., Dixon, C., D.-Jafari, J., Walker, V.K., & Palmer, J.D., *Seven newly discovered intron positions in the triose-phosphate isomerase gene: evidence for the introns-late theory*, Proc. Natl. Acad. USA **92** (1995), 8507-8511.
15. Noguti, T., & Gō, M. *Tracing biological evolution in protein and gene structures.* eds., Gō, M., & Schimmel, P., Elsevier, New York, 1995, pp. 161-174.
16. Stoltzfus, A., Spencer, D.F., Zuker, M., Logsdon, J.M. Jr., & Doolittle, W.F., *Testing the exon theory of genes: the evidence from protein structure*, Science **265** (1994), 202-207.
17. Fedorov, A., Suboch, G., Bujakov, M., & Fedorova, L., *Analysis of nonuniformity in intron phase distribution*, Nucl. Acid Res. **20** (1992), 2553-2557.
18. Long, M.Y., Rosenberg, C., & Gilbert, W., *Intron phase correlations and the evolution of the intron/exon structure of genes*, Proc. Natl. Acad. Sci. USA **92** (1995), 12495-12499.
19. Ambroziak, W., & Pietruszko, R. in *Enzymology and molecular biology of carbonyl metabolism 4*, eds. Weiner, H., Crabb, D.W., & Flynn, T.G., Plenum, New York, 1993, pp. 5-15.
20. Harrington, M.C., Henehan, G.T.M. & Tipton, K.F., *The roles of human aldehyde dehydrogenase isozymes in ethanol metabolism.* Prog. Clin. Biol. Res. **232** (1987), 111-125.
21. Jakoby, W.B. & Ziegler, D.M., *The enzymes of detoxification*, J. Biol. Chem. **265** (1990), 20715-20718.
22. Hsu, L.C., Chang, W.-C., & Yoshida, A., *Genomic structure of the human cytosolic aldehyde dehydrogenase gene*, Genomics **5** (1989), 857-865.
23. Hsu, L.C., Bendel, R.E., & Yoshida, A., *Genomic structure of the human mitochondrial aldehyde dehydrogenase gene*, Genomics **2** (1988), 57-65.
24. Hsu, L.C., Chang, W.-C., Shibuya, A., & Yoshida, A., *Human stomach aldehyde dehydrogenase cDNA and genomic cloning, primary structure, and expression in* Escherichia coli, J. Biol. Chem. **267** (1992), 3030-3037.
25. Hu, C.A., Lin, W.W., & Valle, D., *Cloning, characterization and expression of cDNAs encoding human Δ 1'-pyrroline-5-carboxylate dehydrogenase*, J. Biol. Chem. **271** (1996), 9795-9800.
26. Hsu, L.C., Chang, W.-C., Hiraoka, L.R., & Hsieh, C.L., *Molecular cloning, genomic organization, and chromosomal organization of additional human aldehyde dehydrogenase gene,* ALDH6, Genomics **24** (1994), 333-341.
27. Hsu, L.C., Chang, W.-C., & Yoshida, A., *Cloning a cDNA encoding human ALDH7, a new member of aldehyde dehydrogenase family*, Gene **151** (1994), 285-289.
28. Lin, S.W., Chen, J.C., Hsu, C.L., & Yoshida, A., *Human gamma-aminobutyraldehyde dehydrogenase (ALDH9): cDNA sequence, genomic organization, polymorphism, chromosomal localization, and tissue expression*, Genomics **34** (1996), 376-380.
29. De Laurenzi, V., Rogers, G.R., Hamrock, D.J., Marekov, L.N., Steinert, P.M., Compton, J., Markova, N., & Rizzo, W.B., *Sjogren-Larsson syndrome is caused by mutations in the fatty aldehyde dehydrogenase gene*, Nature Gen. **12** (1996), 52-57.
30. Rost, B., & Sander, C., *Prediction of protein secondary structure at better than 70% accuracy*, J. Mol. Biol. **232** (1993), 584-599.
31. Rost, B., & Sander, C., *Conservation and prediction of solvent accessibility in protein families*, Proteins, **20** (1994), 216-226.
32. Takezaki, N., Rzhetsky, A., & Nei, M., *Phylogenetic test of the molecular clock and linearized trees*, Mol. Biol. Evol. **12** (1995), 823-833.
33. Hedges, S.B., Parker, P.H., Sibley, C.G., & Kumar, S., *Continental breakup and the ordinal diversification of birds and mammals*, Nature (London) **381** (1996), 226-229.
34. Knoll, A.H., *End of the Proterozoic Eon*, Sci. Am. **265** (1991), 64-73.
35. Saitou, N., & Nei, M., *The neighbor-joining method: a new method for reconstructing phylogenetic trees*, Mol. Biol. Evol. **4** (1987), 406-425.
36. Kumar, S., Tamura, K., & Nei, M., *MEGA: Molecular Evolutionary Genetics Analysis*, 1993, The Pennsylvania State University, University Park, PA, Version 1.0.
37. Zuckerkandl, E., & Pauling, L. in *Evolving Genes and Proteins*, eds. Bryson, V., & Vogel, H. J., Academic Press, New York, 1965, pp. 97-166.

38. Han, T.-M., & Runnegar, B., *Megascopic eukaryotic algae from the 2.1-billion-year-old Negaunce iron-formation, Michigan* Science **257** (1992), 232-235.
39. Felsenstein, J., *Confidence limits on phylogenies: an approach using the bootstrap*, Evolution **39** (1985), 783-791.
40. Keilson, J., *Markov chain models - rarity and exponentiality* , Springer-Verlag, New York, 1979.
41. Felsenstein, J., *Evolutionary trees from DNA sequences: a maximum likelihood approach* , J. Mol. Evol. **17** (1981), 368-376.
42. Akaike, H., *A new look at the statistical model identification*, IEEE Trans. Autom. Contr. **AC-19** (1974), 761-723.
43. Martinez, P., Martin, W., & Cerff, R., *Structure, evolution and anaerobic regulation of a nuclear gene encoding cytosolic glyceraldehyde-3-phosphate dehydrogenase from maize*, J. Mol. Biol. **208** (1989), 551-565.
44. Fink, G.R., *Pseudogenes in yeast?* Cell **49** (1987), 5-6.
45. Jarrell, K.A., *Inverse splicing of a group II intron*, Proc. Natl. Acad. Sci. USA **90** (1993), 8624-8627.
46. Nugent, J.M., & Palmer, J.D., *RNA-mediated transfer of the gene coxII from the mitochondrion to the nucleus during flowering plant evolution* , Cell **66** (1991), 473-481.
47. Cech, T.R. *Five easy pieces*, Cell **72** (1986), 161-164.
48. Weiner, A.M., *mRNA splicing and autocatalytic introns: distant cousins or the products of chemical determinism?* Cell **72** (1993), 161-164.
49. Ferat, J.-L., & Michel, F., *Group II self-splicing introns in bacteria,* Nature (London) **354** (1993), 358-361.
50. Zimmerly, S., Guo, H., Perlman, P.S., & Lambowitz, A.M., *A group II intron RNA is a catalytic component of a DNA endonuclease involved in intron mobility*, Cell **82** (1995), 545-554.
51. Yang, J., Zimmerly, S., Perlman, P.S., & Lambowitz, A.M., *Efficient integration of an intron RNA into double-stranded DNA by reverse splicing* , Nature (London) **381** (1996), 332-335.
52. Fukami-Kobayashi, K., Mizutani, M., & Gō, M. (1995) in *Tracing biological evolution in protein and gene structures.* eds., Gō , M., & Schimmel, P. , Elsevier, New York, pp. 271-282.
53. Kladde, M.P., & Simpson, R.T., *Positioned nucleosomes inhibit* Dam *methylation* in vivo, Proc. Natl. Acad. Sci. USA **91** (1994), 1361-1365.

Andrey Rzhetsky
Current address: COLUMBIA GENOME CENTER, COLUMBIA UNIVERSITY, 630 WEST 168TH STREET -BB 16-1611, NEW YORK, NY 10032
E-mail address: andrey@genome2.cpmc.columbia.edu

Francisco José Ayala
Current address: INSTITUTE OF MOLECULAR EVOLUTIONARY GENETICS AND DEPARTMENT OF BIOLOGY, PENNSYLVANIA STATE UNIVERSITY, UNIVERSITY PARK, PA 16802
E-mail address: zeayala@psu.edu

Lily C. Hsu, Cheng Chang, Akira Yoshida
Current address: DEPARTMENT OF BIOCHEMICAL GENETICS, BECKMAN RESEARCH INSTITUTE OF THE CITY OF HOPE, DUARTE, CA 91010
E-mail address: ayoshida@smtplink.coh.org, lchsu@smtplink.coh.org.

This page intentionally left blank

https://doi.org/10.1090/dimacs/037/07

DIMACS Series in Discrete Mathematics
and Theoretical Computer Science
Volume **37**, 1997

Dissimilarity Maps and Substitution Models: Some New Results

Vincent Moulton, Mike Steel, and Chris Tuffley

ABSTRACT. In part one we describe some new results on reconstructing trees from distance measures, based on results in Peter Buneman's pioneering (1971) paper. In part two we analyse a covarion-style model, of the type suggested by Walter Fitch (and colleagues) in 1971 that offers a plausible explanation for why sequence sites "appear" to evolve at different rates.

INTRODUCTION

This paper summarises some new results. Further details (and most of the proofs) will appear elsewhere (see [**28, 32**]). The paper is in two parts:

Part One: Trees with positively-weighted edges induce a natural metric on any subset of vertices, however not every metric is representable in this way. A problem arising in areas of classification, particularly in evolutionary biology, is how to approximate an arbitrary distance function by such a tree metric, and thereby estimate the underlying tree that generated the data. Such transformations, from distances to tree metrics (and thereby to edge-weighted trees) should have some basic properties such as continuity, so that a small change in the input data does not result in a drastically different tree, but this is lacking in several popular methods, for example (as pointed out by Buneman) in methods that attempt to find a closest fit tree metric, and (as we show) in the popular *neighbor joining* method. However known continuous transformations, such as Buneman's original construction, often produce uninteresting (unresolved) trees, which led Buneman to suggest that perhaps this was "the price paid for continuity". One way to extract more information (and continuously!) from distances is Bandelt and Dress' elegant *split decomposition* theory. Based on a modification of Buneman's construction,

1991 *Mathematics Subject Classification.* Primary: 92B10; Secondary: 05C05, 60J27, 92D15, 92D20.

Key words and phrases. Trees, tree metrics, isolation index, nucleotide substitution, covarion model, Markov processes, moment generating function.

The first author was supported in part by the NZ Lotteries Commission.

Correspondence should be directed to the second author.

Work on Part Two was funded by the New Zealand Marsden Fund, contract UOC 516. Thanks also to Dr David Penny, Dr Walter Fitch and Dr Boris Mirkin for their helpful comments.

this continuous map produces a much larger number of splits than Buneman's map, though these splits generally do not form a tree. Here we suggest an alternative modification to the Buneman construction that always leads to trees, and which are, in general, more resolved than those obtained via Buneman's construction. Yet we can achieve this goal without sacrificing continuity. This suggests the possibility of finding other such maps.

Part Two: A "covarion" model for nucleotide substitution which allows sites to turn "on" and "off" with time was proposed 25 years ago by Fitch and Markowitz. It has been argued that evidence supports such models over later, alternative models which postulate a static distribution of rates across sites. However, in contrast to these latter well-studied models, little is known about the analytic properties of the former model. Here we analyse a covarion-style model and show (i) how to obtain the evolutionary distance between two species from the expected proportion of sites where two species differ (ii) that the covarion model cannot be distinguished from a suitably chosen rates-across-sites model on pairs of taxa if only the trace of the joint probability matrix is considered (i.e. the probability that the two taxa are in the same state) and give conditions under which the two models may be distinguished if the full matrix is examined, (iii) that the two models can, in principle, be distinguished when there are at least four monophyletic groups of species. In particular, with a view to a possible test of the covarion hypothesis (against a rates-across-sites model) we construct a distance measure which is a tree metric under certain versions of the covarion model (satisfying a certain separability condition) but which, in general, will not be a tree metric under a rates-across-sites model. Such a measure may also be useful for reconstructing the tree on the monophyletic groups when the covarion model applies.

PART ONE: DISSIMILARITY MAPS

1. Tree metrics, edge-weighted S-trees and indices

Let $S := \{1, \ldots, n\}$, and define

$$\mathcal{D}(S) := \{d : S \times S \to \mathbb{R}_{\geq 0} \ : \ d_{xy} = d_{yx},\ d_{xx} = 0 \text{ for all } x, y \in S\}$$

to be the set of *distance functions* on S. A distance function which satisfies the triangle inequality ($d_{xy} \leq d_{xz} + d_{zy}$ for all $x, y, z \in S$) is said to be a *pseudo-metric*. Endow $\mathcal{D}(S)$ with the l^p norm, that is, set

$$\|d - d'\|_p = \begin{cases} (\Sigma_{i,j} |d_{ij} - d'_{ij}|^p)^{\frac{1}{p}} & p = 1, 2, \ldots \\ \max_{i,j} |d_{ij} - d'_{ij}| & p = \infty. \end{cases}$$

A distance function d on a finite set S is said to be a *tree metric* if there exists a tree $T = (V, E)$, a map $L : S \to V$, called a *labelling*, and a map $w : E \to \mathbb{R}_{>0}$, called an *edge weighting*, such that for all $x, y \in S$, d_{xy} is the sum of $w(e)$ over all edges e in the unique path in T connecting vertices $L(x)$ and $L(y)$.

We may assume that the tree T has no vertices in $V - L(S)$ of degree less than or equal to two, since, as is easily seen, any tree metric on S can be realized by such a tree with a suitable edge weighting. We call such a tree T (together with its associated labelling L) an *S-tree*. S-trees and tree metrics arise in many contexts, particularly in phylogenetic analysis in evolutionary biology (see, for example, [**2, 20**]).

Two fundamental results concerning the characterisation of tree metrics and their uniqueness of representation date back to the 1960s and work by the Russians Zaretsky [**36**] and Smolensky [**30**], and we recall these results and their recent extensions here. One classical result is that a tree metric can arise from only one triple (T, L, w) where T is an S-tree, and w is an edge weighting of T [**1, 6, 30, 36**]. Thus tree metrics are in a natural bijective correspondence with positively edge-weighted S-trees, and, furthermore, there exist fast algorithms for recovering the triple (T, L, w) from d (see, for example, [**1, 2, 19**]). We refer to T (with its associated labelling L) as the S-tree *associated with* d.

Given $d \in \mathcal{D}(S)$ and $\delta \geq 0$, d is said to be *δ-hyperbolic* if

$$d_{ij} + d_{kl} \leq \max\{d_{ik} + d_{jl}, d_{il} + d_{jk}\} + \delta$$

for all $i, j, k, l \in S$. This is a relaxation of the *four-point condition*, in which $\delta = 0$ (for a discussion of this point see [**9**]). A second classical result states that a pseudo-metric d is a tree metric if and only if d is 0-hyperbolic [**6, 30, 36**]. More generally, a result originally given in [**17**], and which is also described in [**5**], states that if a pseudo-metric d is δ-hyperbolic, then there exists a $d' \in \mathcal{T}(S)$ with

$$\|d - d'\|_\infty \leq (1 + \log_2 n)\delta,$$

where $n = |S|$. Thus, if δ is small, then d is close to a tree metric up to a term that grows slowly in n. For a different generalisation of the 0-hyperbolic result to "distance" functions taking values in a suitably structured Abelian monoid $\mathcal{A}$ (and the realisation of such functions by S-trees with edges weights in $\mathcal{A} \setminus \{0\}$) see [**4**].

Let $\mathcal{T}(S)$ be the subspace of $\mathcal{D}(S)$ consisting of tree metrics, and $\mathcal{S}(S)$ be the set of *splits* of S, that is, bipartitions of S. Note that each edge of an S-tree induces a split of S defined by the two non-empty subsets of S that label the two subtrees of T when e is deleted. We say that this split is a *split of* T and is *associated* to edge e. Notice also that any tree metric $d \in \mathcal{T}(S)$ can be conveniently written in the form

$$d = \Sigma_{\sigma \in \mathcal{S}(S)} \lambda_\sigma \cdot \delta_\sigma, \tag{1.1}$$

where

$$\lambda_\sigma = \lambda_\sigma(d) := \begin{cases} w(e) & \text{if } \sigma \text{ is associated to } e \\ 0 & \text{if } \sigma \text{ is not associated to any edge of } T, \end{cases}$$

and where

$$\delta_\sigma(i, j) := \begin{cases} 1 & \text{if } \sigma \text{ separates } i \text{ and } j \\ 0 & \text{otherwise} \end{cases}$$

($\sigma = \{A, B\}$ *separates* i and j if $i \neq j$, and $|\{i, j\} \cap A| = 1$).

Given $d \in \mathcal{D}(S)$ we can define some maps from $\mathcal{S}(S)$ into $\mathbb{R}$, which we call *indices*. We adopt the convenient shorthand xy for d_{xy}. Suppose that $\sigma = \{A, B\}$ is a split of S. Let

$$\mu_\sigma = \mu_\sigma(d) := \frac{1}{2} \cdot \min_{a,a' \in A, b,b' \in B} \{\min\{ab + a'b', ab' + a'b\} - (aa' + bb')\},$$

$$\mu_\sigma^+ = \mu_\sigma^+(d) := \max\{0, \mu_\sigma\},$$

$$\alpha_\sigma = \alpha_\sigma(d) := \frac{1}{2} \cdot \min_{a,a' \in A, b,b' \in B} \{\max\{ab + a'b', ab' + a'b\} - (aa' + bb')\},$$

$$\alpha_\sigma^+ = \alpha_\sigma^+(d) := \max\{0, \alpha_\sigma\}.$$

The map μ is the *Buneman index* [**6**], while α^+ is the *isolation index* [**3**]. Clearly, for any $\sigma \in \mathcal{S}(S)$, we have $\mu_\sigma \leq \alpha_\sigma$ and $\mu_\sigma^+ \leq \alpha_\sigma^+$. The proof of the following lemma can be found in [**3**] and [**6**].

LEMMA 1.1. *If d is an element of $\mathcal{T}(S)$ with $d = \Sigma_\sigma \lambda_\sigma \cdot \delta_\sigma$, then $\lambda_\sigma = \mu_\sigma^+ = \alpha_\sigma^+$ for all $\sigma \in \mathcal{S}(S)$.*

Let $\lambda(d)$ be the vector $[\lambda_\sigma(d)]$ which lies in $\mathbb{R}^{|\mathcal{S}(S)|}$,

$$\mathcal{W}(S) := \{\lambda(d) : d \in \mathcal{T}(S)\},$$

and endow $\mathcal{W}(S) \subseteq \mathbb{R}^{|\mathcal{S}(S)|}$ with the l^p norm. The l^1 norm on the the space $\mathcal{W}(S)$ was proposed in [**29**] as a natural metric for comparing edge-weighted trees. The following theorem shows that $\mathcal{W}(S)$ and $\mathcal{T}(S)$ are homeomorphic. In particular the question of whether or not a map of $\mathcal{D}(S)$ into $\mathcal{T}(S)$ is good does not depend on whether we view the output as a distance function or as an edge-weighted S-tree. The second inequality in Theorem 1.2 is also established, using a slightly different approach, in [**10**, Lemmas 6,7].

THEOREM 1.2. *For $d, d' \in \mathcal{T}(S)$, we have*

$$\begin{aligned} \|d - d'\|_\infty &\leq \|\lambda(d) - \lambda(d')\|_1, \\ \|\lambda(d) - \lambda(d')\|_\infty &\leq 2 \cdot \|d - d'\|_\infty, \end{aligned}$$

and both of these inequalities can be equalities for any S.

PROOF. Writing d, d' in the form of equation (1.1) we have

$$\begin{aligned} \|d - d'\|_\infty &= \max_{i,j} |d_{ij} - d'_{ij}| \\ &= \max_{i,j} |\Sigma_{\{\sigma \in \mathcal{S}(S)\}} (\lambda_\sigma - \lambda'_\sigma) \cdot \delta_\sigma(i,j)| \\ &\leq \max_{i,j} \Sigma_{\{\sigma \in \mathcal{S}(S)\}} |\lambda_\sigma - \lambda'_\sigma| \cdot \delta_\sigma(i,j) \\ &\leq \Sigma_{\{\sigma \in \mathcal{S}(S)\}} |\lambda_\sigma - \lambda'_\sigma| \cdot \max_{i,j} \{\delta_\sigma(i,j)\} \\ &= \|\lambda(d) - \lambda(d')\|_1 \ . \end{aligned}$$

To obtain the second inequality, we show that for any $\sigma \in \mathcal{S}(S)$,

$$|\lambda_\sigma - \lambda'_\sigma| \leq 2 \cdot \delta,$$

where $\delta = \|d - d'\|_\infty$.

Now

$$\begin{aligned} |\lambda_\sigma - \lambda'_\sigma| &= |\mu_\sigma^+(d) - \mu_\sigma^+(d')| \\ &\leq |\mu_\sigma(d) - \mu_\sigma(d')| \\ &\leq 2 \cdot \delta \end{aligned}$$

since, by definition of μ_σ and the triangle inequality

$$\mu_\sigma(d) \leq \mu_\sigma(d') + 2 \cdot \delta,$$

and

$$\mu_\sigma(d') \leq \mu_\sigma(d) + 2 \cdot \delta.$$

This establishes the the two inequalities in the Theorem. To see that they can both be equalities we give the following two examples.

For the first inequality let d be the tree metric induced by the S-tree given by labelling bijectively the degree one vertices of a star tree (a tree having just one

vertex of degree larger than 1) by the elements of S, and assigning weight α to each edge. Let d' be defined in the same way, except that we assign one of the edges weight β instead of α. Then we immediately see that

$$\|d - d'\|_\infty = \|\lambda(d) - \lambda(d')\|_1 = |\alpha - \beta|.$$

For the second inequality, take a tree with four leaves, labelled bijectively by S, and with five edges. Let d be the metric on S induced by assigning weight 2 to all five edges; let d' be the metric on S induced by assigning weight 1 to the central edge and $9/4$ to the other four edges. Then,

$$\|\lambda(d) - \lambda(d')\|_\infty = 1 = 2 \cdot \|d - d'\|_\infty.$$

This completes the proof.

□

2. Retractions

2.1. Preliminaries. A map $\varphi : \mathcal{D}(S) \to \mathcal{D}(S)$ is a *retraction* onto $\mathcal{T}(S)$ if

(i) φ is *continuous*,
(ii) $\varphi(d) \in \mathcal{T}(S)$ for all $d \in \mathcal{D}(S)$, and
(iii) $\varphi(d) = d$ for all $d \in \mathcal{T}(S)$.

Furthermore, if such a retraction φ is *homogeneous*, that is, if

$$\varphi(\lambda d) = \lambda \varphi(d)$$

for all $\lambda > 0$ and $d \in \mathcal{D}(S)$, and if φ is *equivariant*, that is, for all $\tau \in \Sigma_S$ (the permutation group on S)

$$\varphi(d^\tau) = \varphi(d)^\tau,$$

where

$$(d^\tau)_{ij} = d_{\tau(i)\tau(j)},$$

then we say that φ is *good*. These last two properties are desirable in applications in requiring the method to be independent of the units in which d is measured and the names given to the objects in S, respectively [**22, 34**].

Define a partial order on the set of retractions as follows. Given two retractions φ_1, φ_2 of $\mathcal{D}(S)$ onto $\mathcal{T}(S)$, and a metric $d \in \mathcal{D}(S)$, let

$$\varphi_i(d) = \Sigma_{\sigma \in \mathcal{S}(S)} \lambda^i_\sigma(d) \cdot \delta_\sigma, \quad i = 1, 2.$$

We say that φ_2 *refines* φ_1, written $\varphi_1 \preceq \varphi_2$, if and only if for all $d \in \mathcal{D}(S)$ we have

$$\lambda^1_\sigma(d) \leq \lambda^2_\sigma(d),$$

for all $\sigma \in \mathcal{S}(S)$. As can be easily verified, $\preceq$ is a partial order. Note that if $\varphi_1 \preceq \varphi_2$, and if T_1, T_2 are the S-trees associated with $\varphi_1(d)$, $\varphi_2(d)$, respectively, then T_2 is a *refinement* of T_1, in the sense that T_1 can be obtained from T_2 by collapsing a subset of edges.

As has been pointed out (see [**6, 34**]) many early maps for constructing tree metrics fail to be continuous. It can also be shown that the currently popular (in biology) "neighbour-joining" method (see [**20**, p. 488]) is also discontinuous, even when $n = 4$ [**28**].

2.2. The Buneman retraction. Two splits $\sigma = \{A, B\}, \sigma' = \{A', B'\}$ in $\mathcal{S}(S)$ are said to be *compatible* if at least one of the intersections $A \cap A'$, $A \cap B'$, $B \cap A'$, $A' \cap B'$ is empty. If two splits σ, σ' are not compatible then we say that they are *incompatible*, and denote this by writing $\sigma \perp \sigma'$. Clearly any S-tree gives a set of pairwise compatible splits: just take the set of splits induced by the set of edges of the tree. Moreover in [**6**] it is shown that a set of pairwise compatible splits gives rise to a unique tree.

THEOREM 2.1. [**6**] *The set $\{\sigma : \mu_\sigma > 0\}$ is a pairwise compatible collection of splits, and thus gives rise to a unique S-tree.*

The index μ is the basis for the following good map, which is given in [**6**]. We define the *Buneman retraction* $\varphi_B : \mathcal{D}(S) \to \mathcal{T}(S)$ by setting

$$\begin{aligned} \varphi_B(d) &:= \Sigma_{\{\sigma : \mu_\sigma > 0\}} \mu_\sigma \cdot \delta_\sigma \\ &= \Sigma_{\sigma \in \mathcal{S}(S)} \mu_\sigma^+ \cdot \delta_\sigma. \end{aligned}$$

By the previous corollary and the properties of the Buneman index μ, φ_B is a good map. In addition, from [**6**], $\varphi_B(d) \leq d$, in the sense that

$$\varphi_B(d)_{ij} \leq d_{ij}, \text{ for all } i, j \in S.$$

2.3. The Refined Buneman Retraction. In this section we define a new index map $\overline{\mu}_\sigma$ which refines the Buneman index, in the sense that $\overline{\mu}_\sigma \geq \mu_\sigma$ for all $\sigma \in \mathcal{S}(S)$, with strict inequality holding for certain cases. We assume throughout this section that $n \geq 4$.

Writing $q := ab|cd$ to denote the bipartition $\{\{a, b\}, \{c, d\}\}$ of the subset $\{a, b, c, d\}$ of S, let

$$\beta_q := \frac{1}{2}(\min\{ac + bd, ad + bc\} - (ab + cd)).$$

Thus, given a split $\sigma = \{A, B\}$ of S, the Buneman index of σ is given by

$$\mu_\sigma = \min_{a,a' \in A, b,b' \in B} \{\beta_{aa'|bb'}\}.$$

Let Q be the set of $q = aa'|bb'$ consisting of all unordered choices of $a, a' \in A$, and $b, b' \in B$, insisting, furthermore, that if $|A| \geq 2$, then $a \neq a'$ and if $|B| \geq 2$, then $b \neq b'$. Now let $q_1, \ldots, q_{|Q|}$ be an ordering of the elements in Q such that $\beta_{q_i} \leq \beta_{q_j}$ for all $1 \leq i \leq j \leq |Q|$, and define the *refined Buneman index* by

$$\overline{\mu}_\sigma := \frac{1}{n-3} \cdot \Sigma_{i=1}^{n-3} \beta_{q_i}.$$

Note that, by definition, $\overline{\mu}_\sigma \geq \mu_\sigma$ for all $\sigma \in \mathcal{S}(S)$.

LEMMA 2.2 ([**28**]). *If σ and σ' are incompatible splits then*

$$\mu_\sigma + \mu_{\sigma'} \leq 0, \quad \overline{\mu}_\sigma + \overline{\mu}_{\sigma'} \leq 0.$$

This generalises Theorem 2.1, and is a useful tool for establishing the next theorem.

THEOREM 2.3. [**28**] *The map*

$$\psi : d \mapsto \Sigma_{\{\sigma : \overline{\mu}_\sigma > 0\}} \overline{\mu}_\sigma \cdot \delta_\sigma,$$

is a good map, and $\varphi_B \preceq \psi$.

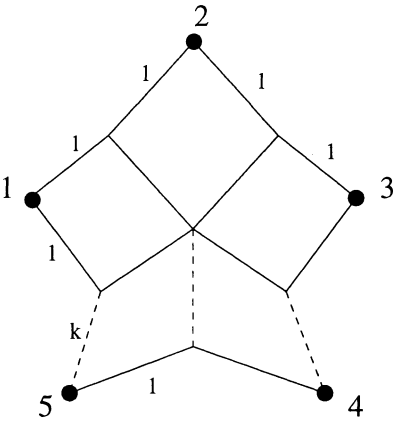


FIGURE 1. *All edges are weighted 1 except dotted edges, which are weighted k.*

We now give a simple example to illustrate that, in certain cases, the refined Buneman retraction gives us a tree which strictly refines the tree given by the Buneman retraction i.e. $\varphi_B \prec \psi$.

Consider the metric d_k on the set $\{1, \ldots, 5\}$ given by the shortest (weighted) path between vertices of the edge-weighted graph in Figure 1, where all edges have weight one except those which are dotted, which have weight k, for some $k \geq 0$.

The Buneman tree for d_k depends upon the value of k. For the case $0 \leq k \leq 2$ the Buneman tree is simply a vertex. If $k \geq 2$, then the Buneman tree consists of one edge of length $k-2$, with its endpoints labelled by $\{1,2,3\}$ and $\{4,5\}$. Thus, in either case, the Buneman tree is highly unresolved (in the sense of [**2**]).

However, in contrast to this, the refined Buneman tree (i.e. that given by using the refined Buneman index), the topology of which also depends upon k, and which is shown in Figure 2, is fully resolved for $k > 0$. Note that in the case where $k = 1$ we get, as might be expected, a star tree.

Note that, in biological applications at least, a desirable feature of a good map is that it be efficiently computable. We will address the computability of the refined Buneman retraction and applications of the refinement to biological data elsewhere [**21**].

2.4. Identifying S-trees using the Buneman retraction and its refinement. We show how the Buneman retraction (or its refinement) essentially identifies the underlying S-tree of a tree metric d' when applied to a distance function d that is close enough to d'. This is summarized in the following theorem.

THEOREM 2.4. *Let $\varphi = \varphi_B$ or ψ (the Buneman retraction or its refinement). Suppose that $d' \in \mathcal{T}(S)$ has associated S-tree T and edge weighting w. Let*

$$x := \min\{w(e) : e \in T\},$$

and suppose that for $d \in \mathcal{D}(S)$ one has

$$\|d - d'\|_\infty < \frac{x}{2}.$$

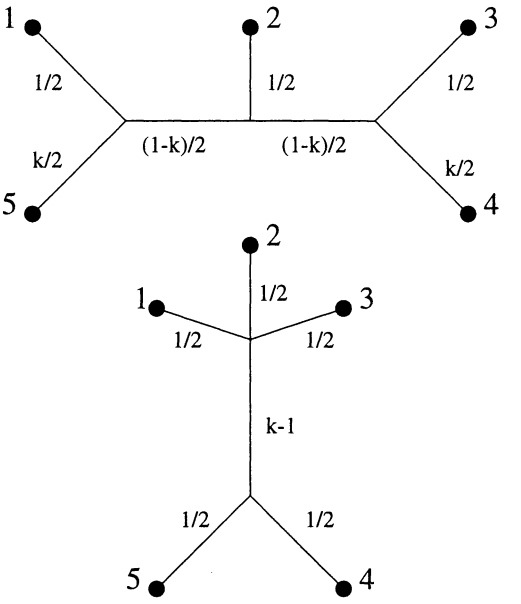


FIGURE 2. *The refined Buneman tree: the top tree is for the case* $0 \leq k \leq 1$ *and the bottom for the case* $1 \leq k$.

Then the S-tree, t, associated to $\varphi(d)$ *refines* T *and the weight of any edge in* t *that does not correspond to an edge of* T *is less than* x. *In particular, if* T *is a fully resolved tree (i.e. every vertex has degree* 1 *or* 3*), then* $t = T$.

PROOF. Suppose $\varphi = \varphi_B$ and let σ be a split of T corresponding to edge e. Then $\mu_\sigma(d') = w(e) \geq x$. Now

$$|\mu_\sigma(d) - \mu_\sigma(d')| \leq 2\delta, \tag{2.1}$$

where $\delta = ||d - d'||_\infty$, as in the proof of Theorem 1.2. Hence, since $\delta < x/2$,

$$\begin{aligned} \mu_\sigma(d) &\geq \mu_\sigma(d') - 2\delta \\ &> x - x = 0, \end{aligned}$$

and so σ is a split of t. Thus t refines T, and in particular, if T is fully resolved then $t = T$.

If T is not fully resolved and σ is a split of t but not T, then by (2.1)

$$\begin{aligned} \mu_\sigma(d) &\leq \mu_\sigma(d') + 2\delta \\ &< 0 + x, \end{aligned}$$

and we deduce that the edge e of t corresponding to σ has weight less than x.

The proof for $\varphi = \psi$ is exactly the same, except that the justification of the analogue of (2.1) is slightly more involved.

□

PART TWO: SUBSTITUTION MODELS

3. The models

In order to accurately reconstruct evolutionary trees and time scales from aligned nucleotide sequences it is helpful to model the mechanism by which the sequences came to differ. Such models can be used to devise new techniques for tree reconstruction and analysis, and also to determine cases where existing methods are likely to lead to erroneous results (see for example [**11**]).

The simplest and earliest models assume that each site evolves i.i.d. at the same rate, and according to simple Markov-style assumptions. However, this single-rate assumption appears to be unrealistic, and accordingly models incorporating some variation of rates across sites have been proposed and studied to take into account different functional constraints at different sites (see for example [**7, 31, 35**]). An alternative approach to accounting for differing selective constraints is Fitch and Markowitz's "concomitantly variable codons" or "covarion" hypothesis [**15**]. This is that at any given time, some sites are invariable due to functional or structural constraints, but that as mutations are fixed elsewhere in the sequence these constraints may change, so that sites that were previously invariable may become variable and vice versa. The pool of variable sites is therefore changing with time (see Figure 3). Since its proposal 25 years ago, it has been argued that evidence supports the covarion hypothesis, both on biochemical grounds, and by providing a better description of certain data [**13, 14, 27**]. However, in contrast to the rates-across-sites models, little is known about the analytic properties of covarion-style models.

Here we present and analyse a simple covarion-style model. Although the motivation for this model clearly says that the i.i.d. assumption is not valid, without it the mathematics becomes much more difficult. We therefore keep this assumption and model the behaviour only of a covarion-style process, with a two-state Markov process that acts as a "switch", turning sites "on" (variable) and "off" (invariable). We do not impose any restrictions on the Markov process that operates at the variable sites other than that it is stationary and reversible. Using techniques from the theory of Markov processes such a model may be analysed and compared with rates-across-sites models in terms of the expected frequencies of site patterns the models should generate.

3.1. A covarion-style model. We model a covarion-style process with two parts: a "switch" process, and an "observable" process, which operates while the switch is "on". Only the state of the observable process, and not that of the switch process, is able to be measured.

The switch is governed by a two state continuous time Markov process with state space $\mathcal{O} = \{\texttt{on}, \texttt{off}\}$ and rate matrix

$$S = \begin{pmatrix} -s_1 & s_1 \\ s_2 & -s_2 \end{pmatrix}$$

where $s_i > 0$ for each i. It is assumed to have the stationary initial distribution $\sigma = (\sigma_1, \sigma_2)$ where

$$\sigma_1 = \frac{s_2}{s_1 + s_2}, \quad \sigma_2 = \frac{s_1}{s_1 + s_2},$$

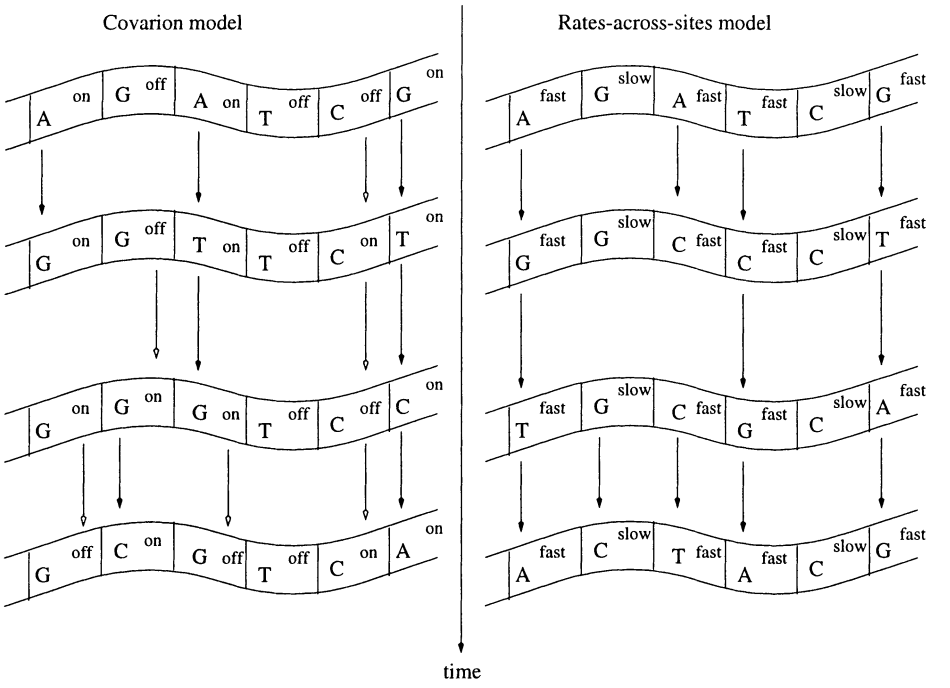


FIGURE 3. Contrasting a covarion style process and rates-across-sites. Under a covarion style process, each site is either "on" or "off". Sites that are off are unable to change character state, but may later turn on (due to character state changes elsewhere in the sequence) and be able to change. Under rates-across-sites, sites evolve at different rates (shown here as "fast" and "slow"), with faster sites changing more frequently than slower ones. The rate at a given site is assumed constant across the entire tree.

so that it is stationary and time-reversible. For a background in Markov processes, the reader is referred to [**16**] and [**24**].

While the switch is in state `off`, the observable process is unable to change state; however, when the switch is in state `on`, the observable process is governed by a second stationary and time reversible Markov process with state space $\mathcal{A} = \{1, \ldots, r\}$, rate matrix R satisfying $R_{ij} > 0$ if $i \neq j$, and initial distribution π. Stationarity and time-reversibility are equivalent to the conditions

$$\pi R = 0 \quad \text{and} \quad \pi_i R_{ij} = \pi_j R_{ji} \quad \text{for all } i, j.$$

In general, for positive integer n we denote the set $\{1, \ldots, n\}$ by $[n]$, and we write $C = (R, S)$ for the covarion model C with observable process rate matrix R and switch process rate matrix S.

This model may be alternatively formulated in terms of a single time-reversible Markov process with state space $\mathcal{A} \times \mathcal{O}$ (which we identify with $[2r]$ according to $(i, \texttt{on}) \mapsto i$, $(i, \texttt{off}) \mapsto i + r$), initial distribution $\pi' = (\sigma_1 \pi, \sigma_2 \pi)$ and $2r \times 2r$ rate

matrix

$$R' = \begin{pmatrix} R - s_1 I_r & s_1 I_r \\ s_2 I_r & -s_2 I_r \end{pmatrix},$$

where I_r denotes the $r \times r$ identity matrix. In this formulation we assume that we are unable to distinguish between the states $(i, \texttt{on})$ and $(i, \texttt{off})$.

It is easily checked that R' is stationary and time-reversible whenever R and S are. Further, both formulations lead to the same joint probability matrix for $\mathcal{A} \times \mathcal{A}$. Using this formulation we may also show that the resulting random process on $\mathcal{A}$ is not in general Markov, so the covarion model may not be analysed by simply treating it as a Markov process on $\mathcal{A}$.

3.2. Rates-across-sites. A *rates-across-sites* model $D = (Q, \mathcal{D})$ consists of a stationary and time-reversible continuous time Markov process with rate matrix Q and initial distribution θ, and a distribution $\mathcal{D}$ of rates ν, which may be either discrete or continuous. We denote the cumulative distribution function of $\mathcal{D}$ by $F_{\mathcal{D}}$.

Each site evolves according to rate matrix νQ where ν is chosen i.i.d. according to $\mathcal{D}$. The rate at a given site is assumed constant across the whole tree. This kind of model has been well studied, see for example [**7, 31, 35**].

4. The two taxa tree

Here we calculate the joint probability matrix for the two taxa tree (that is, the matrix whose ij entry is the probability that taxa 1 is in state i and taxa 2 is in state j), and give conditions under which a suitably chosen rates-across-sites model will agree with a covarion model on all two taxa trees. We also consider the limiting cases of the covarion model as the rate of the switch tends either to zero or to infinity.

4.1. Under the covarion model. Time reversibility implies we may assume the tree is rooted at either of the leaves. Let the process operate for time τ on the edge between the two taxa and write $J_C(\tau)$ for the joint probability matrix. We regard τ as the "length" of the edge. Put $\Pi = \operatorname{diag}(\pi)$ and let $J(t)$ be the joint probability matrix of the unswitched observable process (that is, the Markov process with rate matrix R and initial distribution π operating in the absence of the switch) for time t. If the occupation time of state on in time τ is the random variable $X(\tau)$, then, as far as the observable process is concerned, the edge has effective length $X(\tau)$. The joint probability matrix, given the value of $X(\tau)$, is then $J(X(\tau))$. It follows that

$$J_C(\tau) = \mathbb{E}[J(X(\tau))].$$

Reversibility allows us to obtain a spectral representation of $J(t)$ (see Keilson [**24**, pp. 32–35]). Since ΠR is symmetric so is $\Pi^{1/2} R \Pi^{-1/2}$ which therefore has real eigenvalues $\{\lambda_j\}$ and orthonormal eigenvectors $\{\mathrm{u}_j\}$ (related to the eigenvectors $\{v_j\}$ of R by $v_j = \Pi^{-1/2} u_j$ and $R v_j = \lambda_j v_j$). We then find that

$$J(t) = \sum_{j=1}^{r} e^{\lambda_j t} w_j w_j^{\mathrm{T}},$$

where $w_j = \Pi^{1/2} u_j$ and the superscripted T denotes transposition. Hence

$$\begin{aligned} J_C(\tau) &= \mathbb{E}\left[\sum_{j=1}^{r} e^{\lambda_j X(\tau)} w_j w_j^{\mathrm{T}}\right] \\ &= \sum_{j=1}^{r} \mathbb{E}[e^{\lambda_j X(\tau)}] w_j w_j^{\mathrm{T}}. \end{aligned}$$

From Darroch and Morris [**8**] we have

$$\mathbb{E}[e^{\lambda X(\tau)}] = \sigma^{\mathrm{T}} e^{\tau(S+\lambda D)} \mathbf{1}, \tag{4.1}$$

where $D = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$ and $\mathbf{1} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$, and so, diagonalising $S + \lambda D$ we obtain

$$J_C(\tau) = \sum_{j=1}^{r} [c_j^+ e^{\mu_j^+ \tau} + c_j^- e^{\mu_j^- \tau}] w_j w_j^{\mathrm{T}}, \tag{4.2}$$

where μ_j^+ and μ_j^- are the positive and negative roots of

$$\mu^2 + (s_1 + s_2 - \lambda)\mu - s_2\lambda = 0$$

for $\lambda = \lambda_j$, and

$$c_j^+ = \frac{-(s_1 + s_2 + \mu_j^+)\mu_j^-}{(s_1 + s_2)(\mu_j^+ - \mu_j^-)} \quad \text{and} \quad c_j^- = \frac{(s_1 + s_2 + \mu_j^-)\mu_j^+}{(s_1 + s_2)(\mu_j^+ - \mu_j^-)} \quad .$$

These co-efficients may be shown to satisfy the following inequalities:

LEMMA 4.1.

1. $\lambda_j \leq 0$ *for* $j = 1, \ldots, r$, *with zero occurring as an eigenvalue exactly once.*
2. μ^+ *and* μ^- *are real increasing functions of* λ *satisfying* $\mu^- \leq -(s_1 + s_2) < -s_2 < \mu^+ \leq 0$ *on* $(-\infty, 0]$.
3. $c_j^+, c_j^- \geq 0$ *(with equality only for* $\lambda = 0$, *when* $c^- = 0$*) and* $c_j^+ + c_j^- = 1$.
4. *trace*$(w_j w_j^{\mathrm{T}}) > 0$ *and* $\sum_{j=1}^{r}$ *trace*$(w_j w_j^{\mathrm{T}}) = 1$.

An additional expression of interest is the trace of the joint probability matrix, which is the probability that the two species agree at a given site. Denoting the zero eigenvalue by λ_1, from equation (4.2) we obtain

$$\mathrm{trace}(J_C(\tau)) = \pi\pi^{\mathrm{T}} + \sum_{j=2}^{r} [c_j^+ e^{\mu_j^+ \tau} + c_j^- e^{\mu_j^- \tau}] \mathrm{trace}\left(w_j w_j^{\mathrm{T}}\right). \tag{4.3}$$

4.2. Under rates-across-sites. Put $\Theta = \mathrm{diag}(\theta)$ and let Q have eigenvalues $\{\alpha_j\}$. Arguing as for the covarion model, if $\Theta^{1/2} Q \Theta^{-1/2}$ has orthonormal eigenvectors $\{y_j\}$, then the joint probability matrix $J_D(\tau)$ of the rates-across-sites model D is given by

$$J_D(\tau) = \sum_{i=1}^{r} \mathbb{E}[e^{\alpha_j \nu \tau}] z_j z_j^{\mathrm{T}},$$

where $z_j = \Theta^{1/2} y_j$. We may write this as

$$J_D(\tau) = \sum_{i=1}^{r} M(\alpha_j \tau) z_j z_j^{\mathrm{T}} \tag{4.4}$$

where $M(x) = \mathbb{E}[e^{\nu x}]$ is the *moment generating function* of $\mathcal{D}$, given by the Lebesgue-Stieltjes integral

$$M(x) = \int_0^\infty e^{\nu x} dF_{\mathcal{D}}(\nu).$$

If $F_{\mathcal{D}}$ has a continuous derivative $f_{\mathcal{D}}$ (its probability density function) this is simply

$$M(x) = \int_0^\infty e^{\nu x} f_{\mathcal{D}}(\nu) d\nu,$$

while if $\mathcal{D}$ has only finitely many rates $\nu_1, \ldots, \nu_k$ and $\mathbb{P}[\nu = \nu_i] = p_i$ we have

$$M(x) = \sum_{i=1}^{k} p_i e^{\nu_i x}.$$

As in the covarion case we have

$$\text{trace}(J_{\mathcal{D}}(\tau)) = \theta\theta^{\mathrm{T}} + \sum_{i=2}^{r} M(\alpha_j \tau)\text{trace}(z_j z_j^{\mathrm{T}}). \tag{4.5}$$

4.3. Recovering the evolutionary distance under the two models. Equation (4.4) may be written

$$J_D(\tau) = \Theta M(\tau Q), \tag{4.6}$$

where M is the moment generating function of $\mathcal{D}$ applied to matrices. This expression has the advantage of enabling us to calculate the expected number of substitutions K between the two taxa without requiring knowledge of Q, via

$$K = -\text{trace}\left\{\Theta\left[M^{-1}\left(\Theta^{-1} J_D(\tau)\right)\right]\right\} \tag{4.7}$$

[**18, 33**]. Here M^{-1} is the inverse of the moment generating function, again applied to matrices. This expression gives a tree metric, and since row i of $J_D(\tau)$ sums to θ_i, requires knowledge only of $\mathcal{D}$ to reconstruct the tree from $J_D(\tau)$.

If both Q and $\mathcal{D}$ are known we may express K in terms of just the trace of $J_D(\tau)$ as

$$K = -\text{trace}(\Theta Q) f_D^{-1}(\text{trace}(J_D(\tau))), \tag{4.8}$$

where $f_D(\tau) = \text{trace}(J_D(\tau))$ is given by equation (4.5). Note that f_D^{-1} exists since f_D is monotone decreasing.

The property of (4.4) that allows it to be written in the form (4.6) (namely, M is applied to products of the form $\alpha_j \tau$) does not hold for (4.2), and it appears that a transformation analogous to (4.7) does not exist for the covarion model. However, if R and S are known (or estimated) then, as in (4.8), we may express K in terms of $\text{trace}(J_C(\tau))$ as

$$K = -\text{trace}(\Pi R)\sigma_1 f_C^{-1}(\text{trace}(J_C(\tau))),$$

where $f_C(\tau) = \text{trace}(J_C(\tau))$ is given by equation (4.3). Again f_C is monotone decreasing so f_C^{-1} exists.

Note that in applications, the joint probability matrix (J_C or J_D) is estimated from the observed joint frequency matrix $\hat{J}$. Since J_C and J_D are both symmetric, it is usual practice to take the symmetrised matrix $(\hat{J} + \hat{J}^{\mathrm{T}})/2$ as the estimate.

4.4. Distinguishing between the two models. A question of interest is whether it is possible to distinguish the covarion model from rates-across-sites from pair-wise comparisons of sequences. For a fixed $\tau = \tau_1$, if the rates are distributed according to the distribution of $X(\tau_1)/\tau_1$ then we have $J_C(\tau_1) = J_D(\tau_1)$ for $C = (R, S)$ and $D = (R, \mathcal{D})$, so we cannot tell the covarion model apart from rates-across-sites. However the distribution of $X(\tau)/\tau$ depends on τ which opens the possibility that the models may be distinguished if more than one pair is considered. A partial answer to this question is given by the following results:

THEOREM 4.2.

1. *For any covarion model C there is a rates-across-sites model D such that*
$$trace(J_D(\tau)) = trace(J_C(\tau))$$
for all $\tau \geq 0$.
2. *For a given covarion model $C = (R, S)$, there is a rates-across-sites model $D = (Q, \mathcal{D})$ such that*
$$J_C(\tau) = J_D(\tau)$$
for all $\tau \geq 0$ if and only if R has only one distinct non-zero eigenvalue, in which case $\mathcal{D}$ is a discrete two rate distribution and Q is a scalar multiple of R.
3. *For a given rates-across-sites model $D = (Q, \mathcal{D})$, there is a covarion model $C = (R, S)$ such that*
$$J_D(\tau) = J_C(\tau)$$
for all $\tau \geq 0$ if and only if Q has only one distinct non-zero eigenvalue and $\mathcal{D}$ is a discrete two rate distribution, with both rates greater than zero.

Furthermore stationary and reversible rate matrices with exactly one distinct non-zero eigenvalue and stationary distribution π may be completely characterised as scalar multiples of the matrix
$$R_\pi = \mathbf{1}\pi - I_r$$
where $\mathbf{1} = (1, \ldots, 1)^{\mathrm{T}}$.

It follows from Theorem 4.2 that the trace does not contain enough information to distinguish the covarion model from rates-across-sites models, in the sense that data generated by a covarion model could have been generated by a suitably chosen rates-across-sites model. Note however that parts (ii) and (iii) do not completely answer the question of when the two models may be distinguished on the basis of pairwise comparisons and without knowledge of τ. Firstly, the requirement that the times τ are the same is too strong: if $J_C(\tau) = J_D(f(\tau))$ for all $\tau \geq 0$ for an increasing function f such that $f(0) = 0$ and $f(\tau) \to \infty$ as $\tau \to \infty$ then we cannot distinguish between C and D. Secondly, since we can only ever compare finitely many sequences, it is important to know when we may have $J_C(\tau_i) = J_D(\tau_i')$ for times $\tau_1, \ldots, \tau_k, \tau_1', \ldots, \tau_k'$.

Section 5 gives an alternative approach to distinguishing between the covarion and rates-across-sites models.

We include a proof of Theorem 4.2 part (1). For proofs of the remaining parts of this theorem see [**32**].

PROOF. By (4.3) and Lemma 4.1, $\mathrm{trace}(J_C(\tau))$ has the form

$$\mathrm{trace}(J_C(\tau)) = \pi\pi^{\mathrm{T}} + \sum_{j=2}^{r}[c'^{+}_j e^{\mu^{+}_j \tau} + c'^{-}_j e^{\mu^{-}_j \tau}]$$

where $c'^{+}_j, c'^{-}_j > 0$ and $\sum_{j=2}^{r}[c'^{+}_j + c'^{-}_j] = 1 - \pi\pi^{\mathrm{T}}$. If R has k distinct non-zero eigenvalues we may collect terms in $e^{\mu^{\pm}\tau}$ for each eigenvalue, writing $\mathrm{trace}(J_C(\tau))$ in the form

$$\mathrm{trace}(J_C(\tau)) = a_0 + \sum_{i=1}^{2k} a_i e^{-\nu_i \tau},$$

where $a_i, \nu_i > 0$ for each i and $\sum_{i=0}^{2k} a_i = 1$.

Let $\mathcal{D}$ be the discrete distribution of rates such that

$$\mathbb{P}[\nu = \nu_i] = \frac{a_i}{1 - a_0} \qquad i = 1, \ldots, 2k.$$

Then $\mathcal{D}$ is well-defined, and if $D = (R_\pi, \mathcal{D})$ then by (4.5) and Lemma 4.1,

$$\begin{aligned}
\mathrm{trace}(J_D(\tau)) &= \pi\pi^{\mathrm{T}} + \sum_{j=2}^{r} M(-\tau)\mathrm{trace}(z_j z_j^{\mathrm{T}}) \\
&= \pi\pi^{\mathrm{T}} + M(-\tau)(1 - \pi\pi^{\mathrm{T}}) \\
&= a_0 + (1 - a_0)\sum_{i=1}^{2k} \frac{a_i}{1 - a_0} e^{-\nu_i \tau} \\
&= a_0 + \sum_{i=1}^{2k} a_i e^{-\nu_i \tau} \\
&= \mathrm{trace}(J_C(\tau)).
\end{aligned}$$

□

4.5. Limiting cases. We consider the limiting cases of the covarion model when the switch is very slow ($s_1, s_2 \to 0$) and very fast ($s_1, s_2 \to \infty$), keeping s_1/s_2 (the ratio of "off" sites to "on" sites) constant.

For very slow switches we expect few changes between the states `on` and `off` to occur, so that sites in state `on` will tend to remain in state `on`, and sites in state `off` will tend to remain in state `off`. In the limiting case $s_1, s_2 \to 0$ we expect σ_2 of the sites to be invariant and σ_1 of them to be variable. Calculating this limit we find

$$J_C(\tau) \to \sigma_2 J(0) + \sigma_1 J(\tau)$$

as expected.

For fast switches we expect sites to flip back and forth between `on` and `off` very rapidly, and each spend about the same amount of time in state `on`. Since the expected time in state `on` is $\sigma_1\tau$, in the limiting case $s_1, s_2 \to \infty$ with s_1/s_2 constant we expect

$$J_C(\tau) \to J(\sigma_1 \tau).$$

Calculating this limit we find this is indeed the case.

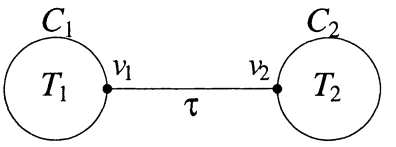


FIGURE 4. The tree joining two monophyletic groups of species C_1 and C_2. The circles marked T_1 and T_2 are rooted subtrees, with roots v_1 and v_2 respectively. The edge $\{v_1, v_2\}$ has length τ.

5. A tree-like measure on monophyletic groups under the covarion model

One approach to testing the covarion model against rates-across-sites models is to examine the sites that are varied and unvaried in two widely separated groups of closely related species. Under the rates-across-sites model, if a given site is in the same state for each member of a group of closely related taxa, then it is likely that the rate of evolution at that site is slow. Since the rate does not change across the tree, we might expect little change to occur in another group of closely related species that is widely separated from the first. On the other hand, under the covarion model if each species has the same state at a given site it seems likely that the site was off for much of the time. In a distant part of the tree the switch might be on so we no longer expect the unvaried sites in the two groups to match up. This observation was made by Fitch [**12**], and examined by Miyamoto and Fitch [**27**], who compared Cu, Zn superoxide dismutase (SOD) sequences from seven mammals and seven plants with simulated sequences generated under covarion and gamma distribution rates-across-sites models, finding that the covarion hypothesis explained the evolution of the protein better than rates-across-sites.

The following discussion is also motivated by Fitch's observation. For a certain class of events and parameters of a covarion model we obtain a tree-like distance measure between monophyletic groups of species that will not in general be tree-like under rates-across-sites models and so could lead to a test of the covarion model against rates-across-sites. The class of covarion models for which this is relevant includes those whose underlying observable process is based on the Kimura [**26**] three-substitution-type model (K3ST) or one of its submodels (the Kimura [**25**] two parameter (K2P) and Jukes-Cantor [**23**] (JC) models).

5.1. Separable events. We describe a class of events that give rise, under the covarion model, to a tree-like metric that is not in general tree-like under a rates-across-sites model.

Suppose E is an event involving an r-state character χ on a set C of species, for example the events

$$E^s = \text{“}\chi(i) \text{ is the same state for all } i \in C\text{”}$$

and

$$E^d = \text{“}\chi(i) \text{ is not the same state for all } i \in C\text{”}.$$

Given two monophyletic groups C_1 and C_2 of species, the tree joining them will be as shown in Figure 4. Let E_i be the event “E occurs for group C_i” for $i = 1, 2$. We

say that the event E is *separable* under the covarion model (R, S) if

$$\mathbb{P}[E_1 \wedge E_2 | \mathsf{0}_1 = \mathsf{o}_1, \mathsf{0}_2 = \mathsf{o}_2] = \mathbb{P}[E_1 | \mathsf{0}_1 = \mathsf{o}_1]\mathbb{P}[E_2 | \mathsf{0}_2 = \mathsf{o}_2] \tag{5.1}$$

for all $\mathsf{o}_1, \mathsf{o}_2 \in \{\mathtt{on}, \mathtt{off}\}$. Note that the separability of a given event may depend on R and S. An analogous condition that might be satisfied by a rates-across-sites model $(Q, \mathcal{D})$ is the following *independence condition:*

$$\mathbb{P}[E_1 \wedge E_2 | \nu] = \mathbb{P}[E_1 | \nu]\mathbb{P}[E_2 | \nu]. \tag{5.2}$$

Let

$$\begin{aligned} p_{12} &= \mathbb{P}[E_1 \wedge E_2], \\ p_i &= \mathbb{P}[E_i], \quad i = 1, 2 \end{aligned}$$

and further in the case of the covarion model let $\mathsf{0}_i$ be the state $\mathtt{on}$ or $\mathtt{off}$ of the switch at the vertex v_i and

$$\begin{aligned} p_i^{\mathtt{on}} &= \mathbb{P}[E_i | \mathsf{0}_i = \mathtt{on}] \\ p_i^{\mathtt{off}} &= \mathbb{P}[E_i | \mathsf{0}_i = \mathtt{off}] \\ \delta_i &= p_i^{\mathtt{on}} - p_i^{\mathtt{off}} \end{aligned}$$

for $i = 1, 2$. Then under conditions (5.1) and (5.2) we have the following:

LEMMA 5.1.

1. *If E is separable under the covarion model (R, S) then*

$$p_{12} - p_1 p_2 = \sigma_1 \sigma_2 e^{-(s_1+s_2)\tau} \delta_1 \delta_2. \tag{5.3}$$

2. *If the independence condition holds for the rates-across-sites model $(Q, \mathcal{D})$ then $p_{12} - p_1 p_2$ does not depend on τ.*

THEOREM 5.2. *For a tree with several monophyletic groups $C_1, \ldots C_n$ at its tips the measure*

$$\rho_{ij} = -\ln |p_{ij} - p_i p_j|$$

is a tree metric realised by the under-lying tree under a covarion model for which E is separable, but in general is not a tree metric under a rates-across-sites model for which the independence condition holds.

PROOF OF LEMMA 5.1 AND THEOREM 5.2. In the covarion case,

$$\begin{aligned} p_{12} &= \sum_{\mathsf{o}_1, \mathsf{o}_2} \mathbb{P}[E_1 \wedge E_2 | \mathsf{0}_1 = \mathsf{o}_1, \mathsf{0}_2 = \mathsf{o}_2]\mathbb{P}[\mathsf{0}_1 = \mathsf{o}_1, \mathsf{0}_2 = \mathsf{o}_2] \\ &= \sum_{\mathsf{o}_1, \mathsf{o}_2} \mathbb{P}[E_1 | \mathsf{0}_1 = \mathsf{o}_1]\mathbb{P}[E_2 | \mathsf{0}_2 = \mathsf{o}_2]\mathbb{P}[\mathsf{0}_1 = \mathsf{o}_1, \mathsf{0}_2 = \mathsf{o}_2], \end{aligned}$$

since E is separable, and

$$p_1 p_2 = \sum_{\mathsf{o}_1, \mathsf{o}_2} \mathbb{P}[E_1 | \mathsf{0}_1 = \mathsf{o}_1]\mathbb{P}[E_2 | \mathsf{0}_2 = \mathsf{o}_2]\mathbb{P}[\mathsf{0}_1 = \mathsf{o}_1]\mathbb{P}[\mathsf{0}_2 = \mathsf{o}_2].$$

Thus

$$p_{12} - p_1 p_2 = \sum_{\mathsf{o}_1, \mathsf{o}_2} \mathbb{P}[E_1 | \mathsf{o}_1]\mathbb{P}[E_2 | \mathsf{o}_2] \left(\mathbb{P}[\mathsf{o}_1, \mathsf{o}_2] - \mathbb{P}[\mathsf{o}_1]\mathbb{P}[\mathsf{o}_2]\right). \tag{5.4}$$

Now the joint probability matrix for the switch operating for time τ is

$$(\mathbb{P}[0_1 = o_1, 0_2 = o_2]) = \sigma_1\sigma_2 \begin{pmatrix} \frac{s_2}{s_1} + e^{-(s_1+s_2)\tau} & 1 - e^{-(s_1+s_2)\tau} \\ 1 - e^{-(s_1+s_2)\tau} & \frac{s_1}{s_2} + e^{-(s_1+s_2)\tau} \end{pmatrix}$$

(see for example [**16**, p. 156]) so the matrix of $\mathbb{P}[0_1 = o_1, 0_2 = o_2] - \mathbb{P}[0_1 = o_1]\mathbb{P}[0_2 = o_2]$ is

$$\sigma_1\sigma_2 e^{-(s_1+s_2)\tau} \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}.$$

Hence, from (5.4),

$$p_{12} - p_1p_2 = \sigma_1\sigma_2 e^{-(s_1+s_2)\tau}(p_1^{\mathtt{on}} - p_1^{\mathtt{off}})(p_2^{\mathtt{on}} - p_2^{\mathtt{off}}) = \sigma_1\sigma_2 e^{-(s_1+s_2)\tau}\delta_1\delta_2$$

as claimed.

Under rates-across-sites with the independence condition holding,

$$\begin{aligned} \mathbb{P}[E_1 \wedge E_2] &= \int_0^\infty \mathbb{P}[E_1 \wedge E_2|\nu]dF_{\mathcal{D}}(\nu) \\ &= \int_0^\infty \mathbb{P}[E_1|\nu]\mathbb{P}[E_2|\nu]dF_{\mathcal{D}}(\nu) \end{aligned}$$

which does not depend on τ, and similarly

$$\mathbb{P}[E_i] = \int_0^\infty \mathbb{P}[E_i|\nu]dF_{\mathcal{D}}(\nu)$$

does not depend on τ, so that $p_{12} - p_1p_2 = \mathbb{P}[E_1 \wedge E_2] - \mathbb{P}[E_1]\mathbb{P}[E_2]$ does not depend on τ either.

Since ρ_{ij} does not depend on the length of the edge between T_i and T_j in the rates-across-sites case, we may rearrange the tree on the groups without changing the value of ρ_{ij}, so the tree on the groups is not uniquely determined by ρ. In the covarion case, if the edge between T_x and T_y has total length τ_{xy} then

$$\begin{aligned} \rho_{xy} &= -\ln|p_{xy} - p_xp_y| \\ &= -\ln(\sigma_1\sigma_2 e^{-(s_1+s_2)\tau_{xy}}|\delta_x||\delta_y|) \\ &= -\ln(\sigma_1\sigma_2) + (s_1+s_2)\tau_{xy} - \ln|\delta_x| - \ln|\delta_y|. \end{aligned}$$

Referring to Figure 5 we have $\tau_{ij} = \tau_i + \tau_j$, $\tau_{ik} = \tau_i + \tau_m + \tau_k$ etc., and Theorem 5.2 follows.

□

We conclude by giving some examples of separable events. Details and results in greater generality appear in [**32**].

THEOREM 5.3 (Some separable events). *The events E^s and E^d above are separable under the covarion model (R, S), and satisfy the independence condition under the rates-across-sites model $(R, \mathcal{D})$, if R has one of the following forms:*

1. *$R = R_K$, where*

$$R_K = \begin{pmatrix} -\delta & \alpha & \beta & \gamma \\ \alpha & -\delta & \gamma & \beta \\ \beta & \gamma & -\delta & \alpha \\ \gamma & \beta & \alpha & -\delta \end{pmatrix}$$

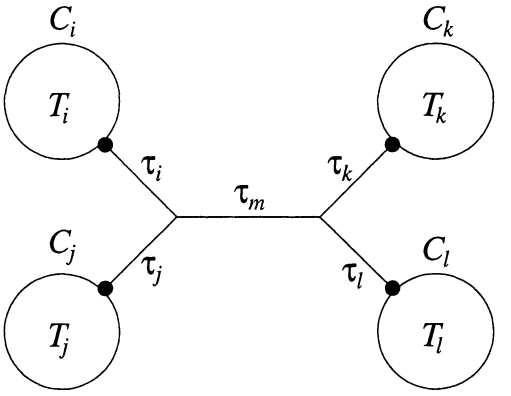


FIGURE 5. The tree on the four monophyletic groups of species C_i, C_j, C_k and C_l. The τ_x are the edge lengths.

and is the form of the matrix used in the K3ST model. This includes as special cases the Kimura two parameter model ($\beta = \gamma$) and the Jukes-Cantor model ($\alpha = \beta = \gamma$).

2. *R is the $r \times r$ matrix given by $R_{ij} = \alpha$ if $i \neq j$ and $R_{ii} = (1 - r)\alpha$, for any r. This gives the fully symmetric model, and includes as particular cases the Cavender-Farris model ($r = 2$) and the Jukes-Cantor model ($r = 4$).*

CONCLUSION

In this paper we have investigated some properties of maps for constructing tree metrics, and contrasted two models of nucleotide substitution. A feature of such models (including the two we described) is that it is often possible to construct distance functions on the sequences which converge in probability to a tree metric (realised by the underlying evolutionary tree), as the sequence length tends to infinity. This is useful because, provided the underlying tree is fully resolved, then any *good* map (as described in Part One) applied to these distance function will reconstruct the correct tree with high probability for sufficiently long sequences. While this might be interpreted as a further advertisement for the virtues of a map being *good*, one in fact requires continuity only at points of $\mathcal{T}(S)$ corresponding to fully resolved trees, and most methods, including neighbor-joining, satisfy this condition. (In case the underlying tree is not fully resolved, then the reconstructed tree will be, with high probability, a refinement of the underlying tree, for sufficiently long sequences, and the maximal weight assigned to any edges that are not in the underlying tree will tend to 0). An important additional issue is the *rate* of convergence of the distance function to an additive metric, and the consequent sequence length required to reconstruct the underlying tree with high probability. Such issues have been addressed by other authors in this book.

References

[1] H.-J. Bandelt, *Recognition of tree metrics*, SIAM J. Discrete Math. **3** (1990), 1–6.

[2] H.-J. Bandelt and A. Dress, *Reconstructing the shape of a tree from observed dissimilarity data*, Advances in Applied Mathematics **7** (1986), 309–343.
[3] ———, *A canonical decomposition theory for metrics on a finite set*, Advances in Mathematics **92** (1992), 47–105.
[4] H.-J. Bandelt and M. A. Steel, *Symmetric matrices representable by weighted trees over a cancellative abelian monoid*, SIAM J. Discrete Math. **8** (1995), no. 4, 517–525.
[5] B. Bowditch, *Notes on Gromov's hyperbolicity criterion for path-metric spaces*, Group Theory from a Geometrical Viewpoint (E. Ghys, A. Haefliger, and A. Verjovsky, eds.), World Scientific Publishing Co. Pte. Ltd., Singapore, New Jersey, London, Hong Kong, 1991, pp. 64–167.
[6] P. Buneman, *The recovery of trees from measures of dissimilarity*, Mathematics in the archaeological and historical sciences (F. R. Hodson, D. G. Kendall, and P. Tautu, eds.), Edinburgh University Press, 1971, pp. 387–395.
[7] J. T. Chang, *Inconsistency of evolutionary tree topology reconstruction methods when substitution rates vary across characters*, Mathematical Biosciences **134** (1996), 189–215.
[8] J. N. Darroch and K. W. Morris, *Passage time generating functions for continuous-time finite Markov chains*, Journal of Applied Probability **5** (1968), 414–426.
[9] A. Dress, V. Moulton, and W. Terhalle, *T-theory – an overview*, The European Journal of Combinatorics **17** (1996), 161–175.
[10] M. Farach and S. Kannan, *Efficient algorithms for inverting evolution*, Proceedings of the 1996 ACM Symposium on the Foundations of Computer Science, 1996.
[11] J. Felsenstein, *Cases in which parsimony or compatibility will be positively misleading*, Systematic Zoology **27** (1978), 401–410.
[12] W. M. Fitch, *The nonidentity of invariable positions in the cytochrome* c *of different species*, Biochemical Genetics **5** (1971), 231–241.
[13] ———, *Rate of change of concomitantly variable codons*, J. Mol. Evol. **1** (1971), 84–96.
[14] W. M. Fitch and F. J. Ayala, *The superoxide dismutase molecular clock revisited*, Proc. Natl. Acad. Sci. USA **91** (1994), 6802–6807.
[15] W. M. Fitch and E. Markowitz, *An improved method for determining codon variability in a gene and its application to the rate of fixation of mutations in evolution*, Biochemical Genetics **4** (1970), 579–593.
[16] G. R. Grimmett and D. R. Stirzaker, *Probability and random processes*, Clarendon Press, Oxford, 1982.
[17] M. Gromov, *Hyperbolic groups*, Essays in Group Theory (S. Gerstin, ed.), MSRI Publ., no. 8, Springer, 1987, pp. 75–263.
[18] X. Gu and W.-H. Li, *A general additive distance with time-reversibility and rate variation among nucleotide sites*, Proc. Natl. Acad. Sci. USA **93** (1996), 4671–4676.
[19] D. Gusfield, *Efficient algorithms for inferring evolutionary trees*, Networks **21** (1991), 19–28.
[20] D. Hillis, C. Moritz, and K. Barbara, *Molecular systematics*, second ed., Sinauer Associates Inc., 1996.
[21] D. Huson, V. Moulton, and M. Steel, *Retractions as a tool for analyzing distance functions in biology*, in preparation.
[22] N. Jardine and R. Sibson, *The construction of hierarchic and non-hierarchic classifications*, The Computer Journal **11** (1968), 177–184.
[23] T. H. Jukes and C. R. Cantor, *Evolution of protein molecules*, Mammalian protein metabolism (H. N. Munro, ed.), Academic Press, New York, 1969, pp. 21–132.
[24] J. Keilson, *Markov chain models—rarity and exponentiality*, Applied Mathematical Sciences, vol. 28, Springer-Verlag, 1979.
[25] M. Kimura, *A simple method of estimating evolutionary rate of base substitutions through comparative studies of nucleotide sequences*, J. Mol. Evol. **16** (1980), 111–120.
[26] ———, *Estimation of evolutionary distances between homologous nucleotide sequences*, Proc. Nat. Acad. Sci. USA **78** (1981), 454–458.
[27] M. M. Miyamoto and W. M. Fitch, *Testing the covarion hypothesis of molecular evolution*, Mol. Biol. Evol. **12** (1995), no. 3, 503–513.
[28] V. Moulton and M. Steel, *Retractions of finite distance functions onto tree metrics*, submitted to SIAM J. Discrete Math.
[29] D. Robinson and L. Foulds, *Comparison of weighted labelled trees*, Combinatorial Mathematics VI, Lecture Notes in Mathematics, vol. 748, Springer, 1979, pp. 119–129.

[30] Y. A. Smolensky, *A method for linear recording of graphs*, USSR Comput. Math. Phys. **2** (1969), 396–397.
[31] M. A. Steel, L. A. Székely, and M. D. Hendy, *Reconstructing trees when sequence sites evolve at variable rates*, Journal of Computational Biology **1** (1994), no. 2, 153–163.
[32] C. Tuffley and M. Steel, *Modelling the covarion hypothesis of nucleotide substitution*, submitted to Mathematical Biosciences, 1996.
[33] P. J. Waddell and M. A. Steel, *General time reversible distances with unequal rates across sites*, Research report 143, Dept. of Mathematics and Statistics, University of Canterbury, New Zealand, 1996.
[34] K. Wolf and P. Degens, *On properties of additive tree algorithms*, Conceptual and Numerical Analysis of Data, Proc. of the 13th conf. of the Gesellschaft für Klasification, Springer-Verlag, 1989.
[35] Z. Yang, *Maximum-likelihood estimation of phylogeny from DNA sequences when substitution rates differ across sites*, Mol. Biol. Evol. **10** (1993), 1396–1401.
[36] K. A. Zaretsky, *Reconstruction of a tree from the distances between its pendant vertices*, Uspekhi Math. Nauk (Russian Mathematical Surveys) **20** (1965), 90–92 (Russian).

BIOMATHEMATICS RESEARCH CENTRE, UNIVERSITY OF CANTERBURY, CHRISTCHURCH, NEW ZEALAND

BIOMATHEMATICS RESEARCH CENTRE, UNIVERSITY OF CANTERBURY, CHRISTCHURCH, NEW ZEALAND
E-mail address: m.steel@math.canterbury.ac.nz

BIOMATHEMATICS RESEARCH CENTRE, UNIVERSITY OF CANTERBURY, CHRISTCHURCH, NEW ZEALAND

This page intentionally left blank

https://doi.org/10.1090/dimacs/037/08

DIMACS Series in Discrete Mathematics
and Theoretical Computer Science
Volume **37**, 1997

The Performance of the Neighbor-Joining Method of Phylogeny Reconstruction

Kevin Atteson

ABSTRACT. Biologists have hotly debated the merits of various methods for phylogenetic reconstruction for many years. In this paper, we analyze the performance of the popular neighbor-joining method. In particular, we determine the L_∞ radius under which the method will determine the correct tree topology. In fact, this radius is optimal for all distance-based methods, the class of methods which includes all known tractable methods for which there is any sort of performance guarantee.

1. Discussion

The phylogenetic reconstruction problem is to determine the evolutionary relationships between a set of species typically from information contained in biomolecular sequence data. These evolutionary relationships may be represented by a phylogenetic tree, that is, a binary tree in which the extant species are represented at the leaves and the internal nodes represent possibly extinct common ancestors of the extant species. Evolutionary relationships are hotly debated among biologists and different relationships are be obtained by different methods. In recent years, the growth of large-scale sequencing has begun to provide a wealth of data for phylogenetic reconstruction.

The many methods available for phylogenetic reconstruction can be classified into distance-based methods and sequence-based methods[**SOWH96**]. All known tractable methods for which there is any kind of performance guarantee are distance-based methods. In this paper, we prove a result which quantifies the performance of one popular method, the neighbor-joining method[**SN87**]. In particular, we prove that this computationally efficient method does best possible, in terms of the L_∞ radius at determining the topology of the tree amongst all distance-based methods.

This paper is related to some other results presented at the workshop. In particular, Olivier Gascuel has presented a new version of the neighbor-joining algorithm. The results of this paper apply to any variant of the neighbor-joining algorithm in which a convex update formula is used (see (2) below) and so apply to the variant of the algorithm presented here by Gascuel as well as other variants. Also, Tandy Warnow has presented a new algorithm for phylogeny reconstruction

1991 *Mathematics Subject Classification.* Primary 68R10,05C05.
Key words and phrases. phylogenetic reconstruction, neighbor-joining.
This work was supported by NSF grant number BIR 9413215.

which, in addition to neighbor-joining, does best possible at finding the correct topology of the tree.

We first present the basic results of the paper and discuss their significance. A distance-based method of phylogenetic reconstruction takes a matrix of distances, D_{ij}, where i and j represent species, and determines a tree T. Now let T be an edge-weighted tree with edge set E and with positive weight $l(e)$ on edge $e \in E$. Let D^T denote the distances between leaves in the tree T, that is:

$$D^T_{ij} = \sum_{e \in P_{i,j}} l(e)$$

where $P_{i,j}$ denotes the set of edges in the path between leaves i and j. A distance matrix D is called additive if there is a weighted tree T such that $D = D^T$. A reasonable distance-based method should return the tree T when given D^T or a distance matrix sufficiently close to D^T as input. However, for any tree T with distance matrix $D = D^T$, there is a distance matrix D' and an additive distance matrix D'' such that D and D'' have distinct topologies but are both distance $\min_{e \in E} \frac{l(e)}{2}$ to D', in the L_∞ norm:

$$\max_{i,j} \left| D_{ij} - D'_{ij} \right| = \min_{e \in E} \frac{l(e)}{2}$$

$$\max_{i,j} \left| D''_{ij} - D'_{ij} \right| = \min_{e \in E} \frac{l(e)}{2}$$

where E is the set of edges of the tree corresponding to D and $l(e)$ is the branch length of the edge $e \in E$. Note that D'' can be chosen to have minimum branch length equal to that of D. See Figure 1 for a graphical representation of this relationship. Hence, no algorithm can always correctly reconstruct tree topologies when the L_∞ (vector) norm between the actual and measured distance matrices is at least $\min_{e \in E} \frac{l(e)}{2}$. In fact, we demonstrate in the following section that the neighbor-joining algorithm will always find the correct tree if this error is less than $\min_{e \in E} \frac{l(e)}{2}$.

By introducing a specific model of the evolution of biomolecular sequences, as in Farach and Kannan[**FK96**], we can determine an upper bound on the sample-size complexity, that is, the number of samples required so that the error in the distances will be within the tolerance necessary for the neighbor-joining algorithm to determine the topology with high probability. The model we introduce is the Cavender-Farris model[**Far73, Cav78**]. Under this model, there is a true rooted tree T with n leaves corresponding to extant observed species and internal nodes corresponding to ancestral species. For any species i (extant or ancestral), there is a sequence of k binary random variables (with only the random variables corresponding to the leaf sequences being observed). The sequence at the root is generated by fair coin flips (i.i.d. uniform). With each edge $e \in E$, we associate a probability $p(e)$, the probability that a given site of the sequence will change along that edge. Each site of the sequence is assumed to mutate in a Markovian fashion with respect to other species, that is, any species is dependent upon any ancestral species (more precisely, any non-descendant species) only through its most recent ancestral species. Each site is assumed to be independent and identically distributed (this

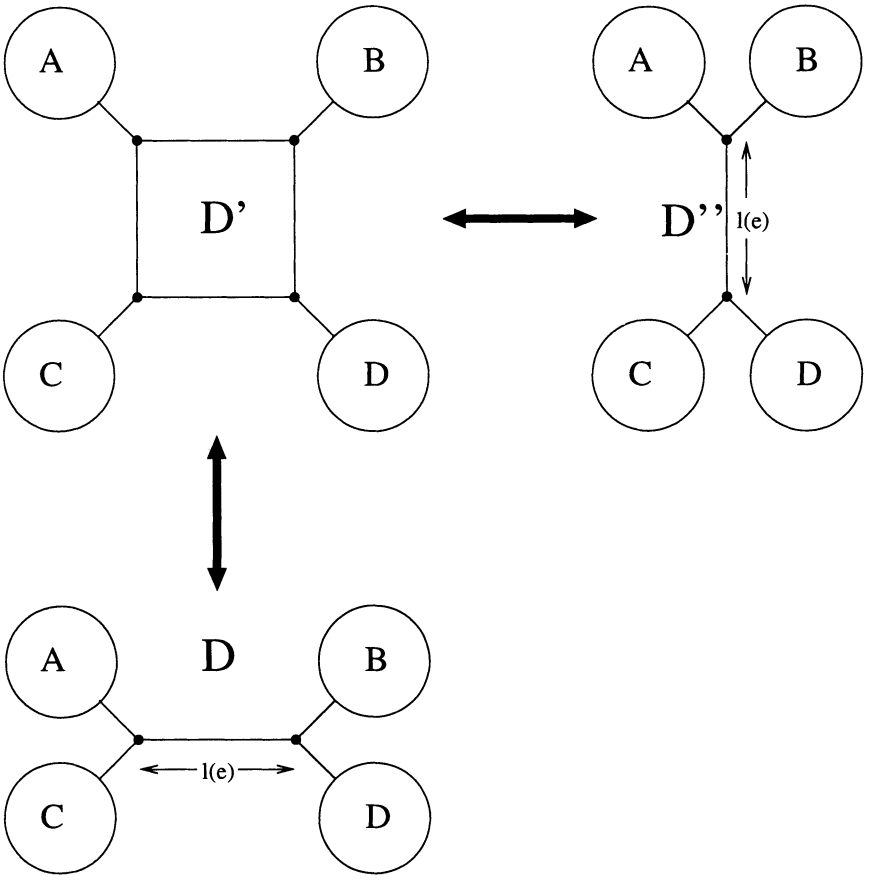


FIGURE 1. A graphical representation of two "nearest" distinct tree topologies. Thanks to Mike Steel for this graphical representation.

is perhaps the biologically most unrealistic assumption of the model). The model generates a sequence of k i.i.d. binary vectors, each having one element for each extant species.

Given the Cavendar-Farris model, let $p_{i,j}$ be the probability of mutation at a given site between extant species i and j. The distance matrix given by:

$$D_{ij} = -\ln(1 - 2p_{i,j})$$

is additive. Letting $\hat{p}_{i,j}$ denote the number of observed mutations which occur between species i and j, we can estimate D_{ij} by:

$$\hat{D}_{ij} = -\ln(1 - 2\hat{p}_{i,j})$$

In fact, using the Azuma-Hoeffding inequality[**GS92**], we can guarantee that $\max_{i,j} \left| D_{ij} - \hat{D}_{ij} \right| < \epsilon$ with probability at least δ if:

$$k \geq \frac{2\ln\left(\frac{2n}{\delta}\right)}{(1-\exp(-\epsilon))^2}\left(\exp\left(\max_{i,j} D_{ij}\right)\right)^2$$

where n is the number of species and k is the required number of observed mutation sites. Hence, if we allow ϵ to equal $\min_{e\in E}\frac{l(e)}{2}$, we can guarantee that the neighbor-joining algorithm will find the correct topology with probability at least δ if k sites are observed. Farach and Kannan[**FK96**] suggest an algorithm which finds a tree which is a 3-approximation of the nearest additive distance matrix using the L_∞ norm to the observed distances, that is, a tree whose distance to the observed tree is at most 3 times as much as the nearest tree to the observed distances. An algorithm which finds the nearest additive distance matrix would find the correct topology when the distances are within $\min_{e\in E}\frac{l(e)}{4}$ of the distances of the true tree. However, this problem is NP-hard to approximate within $\frac{9}{8}$[**ABF+96**]. With Farach and Kannan's algorithm, the correct topology is guaranteed be found if the observed distances are within $\min_{e\in E}\frac{l(e)}{8}$ of the actual distances. Hence, the number of samples needed to be able to give a proven guarantee of some fixed probability of determining the correct topology is up to 16 times as few for the neighbor-joining algorithm than for the algorithm of Farach and Kannan. Note, however, that the number of samples needed to for these guaranteed performance rates is exponential in the diameter of the tree and so may not be practical in many situations of interest.

2. Proof of Results

Let L denote the set of extant taxa. For the purpose of this paper, we say that two taxa $i, j \in L$ are neighbors in a tree T if and only if $|P_{i,j}| = 2$, that is, if there are exactly two edges on the path in the tree between the taxa i and j. The basic idea behind the neighbor-joining algorithm is to attempt to find a pair of species i and j which are neighbors in the tree, modify the distances so as to combine i and j into a single species u and repeat.

The pair of neighbors i and j are chosen to minimize:

$$S_{i,j} = (|L|-2)D_{ij} - \sum_k D_{ik} - \sum_k D_{jk} \tag{1}$$

After combining i and j to form u, the distances are updated in the following manner:

$$D_{uk} = \lambda_u D_{ik} + (1-\lambda_u) D_{jk} \tag{2}$$

with distances between all other taxa remaining unchanged. Originally, Saitou and Nei suggest using $\lambda_u = \frac{1}{2}$ for all u but we consider the general case here where $0 \leq \lambda_u \leq 1$. Another version of the algorithm is given in [**SK88**] but these versions are proved equivalent in [**Gas94**].

A distance matrix will be called additive if there is a tree such that the distance between any two species within the tree, that is, the sum of the edge lengths of the path between the species, corresponds with that distance in the matrix. Our first lemma says that if we combine a pair of neighbors by the updating step of the algorithm on additive distances, the result is the distances of the original tree with

the pair of neighbors replaced by a single leaf hanging off of the node adjacent to the pair of neighbors. A similar result for Sattath and Tversky's predecessor[**ST77**] of Saitou and Nei's neighbor-joing algorithm was proved by Bandelt and Dress[**BD86**].

LEMMA 1. *Let D be an additive distance matrix with a corresponding binary tree T. Fix neighbors i and j in T. Let u' denote the internal vertex adjacent to i and j and u'' the vertex adjacent to u' but different from i and j. The distance matrix on the set of taxa $(L - \{i,j\}) \cup \{u\}$ with distances given by formula (2) is additive and corresponds to the binary created by removing i, j and u' from T and adjoining a new leaf u adjacent to u'' at distance:*

$$l(\{u', u''\}) + \lambda_u l(\{i, u'\}) + (1 - \lambda_u) l(\{j, u'\})$$

PROOF. The proof is straightforward. Denote the graph formed by deleting i, j and u' and adjoining u adjacent to u'' by T'. We first show that T' is a binary tree. For any leaves, k and l, of T other than i and j, there is a path from k to i. This path must go through u''. Hence, there will be a path between k and l in the resulting graph. Similarly, taking the path between k and i, removing u' and i and adding u yields a path between k and u. Hence, the graph is a tree. In fact, it is a binary tree. If u'' is internal then it has degree 3 and so, removing the edge $\{u', u''\}$ and adding edge $\{u, u''\}$ leaves the degree as 3. Otherwise, the resulting tree contains only u and u'' and so is binary.

We must now show that the distances in this tree are equal to the distances given by formula (2). Let $D_{kl}^{T'}$ denote the distances between leaves k and l of T' (we continue to use D as the distances of the original tree and as those given by the formula since there is no ambiguity). Clearly, if k and l are not u, we have that $D_{kl}^{T'} = D_{kl}$ since neither the construction of T' does not affect the path between k and l. Now let $D_{ku''}$ denote the distance between k and u'' in T. We have:

$$\begin{aligned} D_{ku}^{T'} &= D_{ku''} + l(\{u', u''\}) + \lambda_u l(\{i, u'\}) + (1 - \lambda_u) l(\{j, u'\}) \\ &= \lambda_u (D_{ku''} + l(\{u', u''\}) + l(\{i, u'\})) \\ &\quad + (1 - \lambda_u)(D_{u'u''} + l(\{u', u''\}) + l(\{j, u'\})) \\ &= \lambda_u D_{ik} + (1 - \lambda_u) D_{jk} \end{aligned}$$

Hence, $D_{ku}^{T} = D_{ku}$ and so the lemma holds. □

Note that the neighbor selection criterion, formula (1), is linear in the distances which are linear in the edge weights when D is additive. We now present a lemma which determines the weights of this formula on each edge. For a tree T and an edge e, we let $L_k(e)$ denote the set of leaves in the component of $T - e$ containing k (see Figure 2).

LEMMA 2. *Suppose that the distances D are formed from an additive tree T, with edge set E. We have:*

$$S_{i,j} = \sum_{e \in E} w_e(i,j) l(e) \tag{3}$$

where:

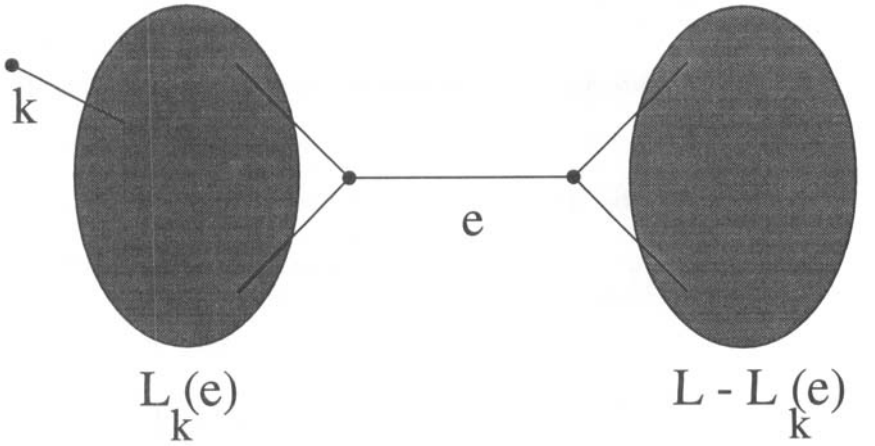


FIGURE 2. An illustration of the notation "$L_k(e)$". "$L_k(e)$" denotes the set of leaves in the component of $T - e$ containing the leaf k.

$$w_e(i,j) = \begin{cases} -2 & \text{if } e \in P_{i,j} \\ -2|L - L_i(e)| & \text{otherwise} \end{cases} \tag{4}$$

where recall that $P_{i,j}$ denotes the set of edges on the path between i and j and L denotes the set of taxa.

Note that $w_e(i,j)$ is symmetric in i and j since for $e \in E - P_{i,j}$, we have that $|L - L_i(e)| = |L - L_j(e)|$.

PROOF. First note that for taxa k and l, the distance D_{kl} is the sum of the branch lengths of the edges in $P_{k,l}$:

$$D_{kl} = \sum_{e \in P_{k,l}} l(e)$$

Hence:

$$\begin{aligned} S_{i,j} &= (|L| - 2)D_{ij} - \sum_k D_{ik} - \sum_k D_{jk} \\ &= \sum_{e \in P_{i,j}} (|L| - 2)l(e) - \sum_k \sum_{e \in P_{i,k}} l(e) - \sum_k \sum_{e \in P_{j,k}} l(e) \end{aligned} \tag{5}$$

Note that $e \in P_{i,k}$ if and only if $k \in L - L_i(e)$. Hence:

$$\sum_k \sum_{e \in P_{i,k}} l(e) = \sum_{e \in E} \sum_{k \in L - L_i(e)} l(e) = \sum_{e \in E} |L - L_i(e)| l(e)$$

Incorporating this into (5) yields:

$$\begin{aligned} S_{i,j} &= \sum_{e \in P_{i,j}} (|L| - 2) l(e) - \sum_{k} \sum_{e \in P_{i,k}} l(e) - \sum_{k} \sum_{e \in P_{j,k}} l(e) \\ &= \sum_{e \in P_{i,j}} (|L| - 2) l(e) - \sum_{e \in E} (|L - L_i(e)| + |L - L_j(e)|)\, l(e) \\ &= \sum_{e \in P_{i,j}} ((|L| - 2) - (|L - L_i(e)| + |L - L_j(e)|))\, l(e) \\ &\quad - \sum_{e \in E - P_{i,j}} (|L - L_i(e)| + |L - L_j(e)|)\, l(e) \end{aligned}$$

However, for $e \in P_{i,j}$, we have that $|L - L_i(e)| + |L - L_j(e)| = |L|$. Also, for $e \in E - P_{i,j}$, we have that $|L - L_i(e)| = |L - L_j(e)|$. Hence:

$$S_{i,j} = -2 \sum_{e \in P_{i,j}} l(e) - 2 \sum_{e \in E - P_{i,j}} |L - L_i(e)| l(e)$$

which was to be shown. □

We now find the difference in $S_{i,j}$ for non-neighbors and neighbors. Towards this end, it would be desirable to determine a lower bound on:

$$\min_{\text{non-neighbors } k,\, l} S_{k,l} - \min_{\text{neighbors } i,\, j} S_{i,j}$$

so that when we consider $S_{k,l}$ calculated using approximate distances, we will be able to determine the tolerance within which the distances can vary. However, it turns out to be easier to bound:

$$\min_{\text{non-neighbors } k,\, l} (S_{k,l} - S_{i,j})$$

for specific neighbors i and j which depend upon k and l. It turns out that using this less strict bound, there is no loss in the tightness of the results.

LEMMA 3. *Let D be additive and the corresponding tree additive. Let S denote the results of formula (1) applied to the distances D. If $k, l \in L$ are not neighbors, then there is a pair of neighbors $i, j \in L$ such that either:*

$$S_{k,l} - S_{i,j} \geq 2(|L| - 3) \min_e l(e)$$

and $\{i, j\} \cap \{k, l\} \neq \emptyset$ or:

$$S_{k,l} - S_{i,j} \geq 3(|L| - 4) \min_e l(e)$$

and $\{i, j\} \cap \{k, l\} = \emptyset$.

PROOF. We first summarize the proof. We consider the subtrees hanging off of the path from k to l and choose i and j to be any pair of neighbors in one of these subtrees which does not uniquely contain a maximal number of leaves. The proof then follows by case analysis. Using Lemma 2, for each $e \in E$, the weight on $l(e)$ in $S_{k,l}$ is at least that in $S_{i,j}$. In particular, for edges which separate i and j from

k and l, the weight in $S_{i,j}$ will be substantially less than the weight in $S_{k,l}$ since the component containing i and j will be smaller than the component containing k and l. Summing bounds on the differences in the weights on various edges leads to the desired result. We now proceed with the details.

Fix any taxa $k, l \in L$ which are not neighbors as in the statement of the lemma. Choose an edge $e^* \in E - P_{k,l}$ which minimizes $|L_k(e^*)|$. Adding e^* to the component of $T - e^*$ which contains k and l produces an unrooted tree with at least 4 leaves (k, l, a vertex of e^* and at least one other which must come off of $P_{k,l}$ since k and l are not neighbors). Every unrooted tree with at least 4 leaves has at least two pairs of neighbors and so this tree has a pair of neighbors which are neighbors in T (that is, a pair which is not incident with e^*). Let i and j be these neighbors (see Figure 4). We show that $w_e(i,j) \leq w_e(k,l)$ and by how much via case analysis on e:

1. Suppose that $e \in P_{k,l}$. We have that $w_e(i,j) = -2|L - L_i(e)| \leq -2 = w_e(k,l)$ since $|L - L_i(e)| \geq 1$ for all e. We now determine a lower bound on the quantity $\sum_{e \in P_{k,l} - P_{i,j}} (w_e(k,l) - w_e(i,j))$ which we have shown is non-negative for the case in which $\{k,l\} \cap \{i,j\} = \emptyset$. Let $P_{k,l} = \{k, v_1, \ldots, v_m, l\}$ with $m \geq 2$ since k and l are not neighbors. Note that each v_p, for $1 \leq p \leq m$ has an edge e_p incident on it which is not in $P_{k,l}$ since the tree is unrooted binary so that every internal node is incident with 3 edges. For some e_{p^*} (see Figure 4), we have $\{i,j\} \in L - L_k(e_{p^*})$ since i and j are neighbors and neither equals k or l. Note that it must be that $L_i(e_{p^*}) \subseteq L_k(e^*)$ since $e_{p^*} \in P_{i,k}$ and $e^* \notin P_{i,k}$. Also, by the minimality of $|L_k(e^*)|$, we have $|L_k(e^*)| \leq |L_k(e_{p^*})|$ and so $|L - L_k(e_{p^*})| = |L_i(e_{p^*})| \leq |L_k(e_{p^*})|$ or $|L_k(e_{p^*})| \geq \frac{|L|}{2}$. Now letting $e' = \{v_{p^*-1}, v_{p^*}\}$ and $e'' = \{v_{p^*}, v_{p^*+1}\}$ (see Figure 4), we have $w_{e'}(i,j) + w_{e''}(i,j) = -2\left(|L - L_i(e')| + |L - L_i(e'')|\right) = -2|L_k(e_{p^*})| \leq -|L|$. Hence, for this pair of edges $w_{e'}(i,j) + w_{e''}(i,j) \leq |L| - 4 + w_{e'}(k,l) + w_{e''}(k,l)$.
2. Suppose $e \in P_{i,k} - P_{k,l}$. We break this case further into two cases depending upon the location of e^*:
   (a) Suppose e^* is in the component of $T - e$ containing i. In this case $L - L_k(e^*) \subset L - L_k(e)$. Note that this is a strict subset since $i \in (L - L_k(e)) - (L - L_k(e^*))$. Hence, $|L_k(e^*)| > |L_k(e)|$ which contradicts the minimality of $|L_k(e^*)|$ among $\{|L_k(e)| : e \in E - P_{k,l}\}$. Hence, this case does not occur.
   (b) Suppose e^* is in the component of $T - e$ containing k. In this case, we have that $L - L_k(e^*) \subset L_k(e)$. By the minimality of $|L_k(e^*)|$ among $\{|L_k(e)| : e \in E - P_{k,l}\}$, we have that $|L| - |L_k(e^*)| \geq |L| - |L_k(e)|$ and so $w_e(k,l) = -2(|L| - |L_k(e)|) \geq -2(|L| - |L_k(e^*)|) > -2|L_k(e)| = -2(|L| - |L_i(e)|) = w_e(i,j)$. We now determine a lower bound on the sum of $w_e(k,l) - w_e(i,j)$ over edges for which this case occurs which we have already shown is positive (assuming there is at least one such edge). Note that since i and j are neighbors and k and l aren't, we can assume without loss of generality that $k \notin \{i,j\}$ (by choosing $l \in \{i,j\}$ if necessary). Hence, there is an edge $e''' \in P_{i,k}$ which separates i and j from the remaining taxa. For this edge, we have that $w_{e'''}(i,j) = -2(|L| - 2)$ and $w_{e'''}(k,l) = -2$ if $l \in \{i,j\}$ and $w_{e'''}(k,l) = -2|L - L_k(e''')| = -4$ if $l \notin \{i,j\}$.

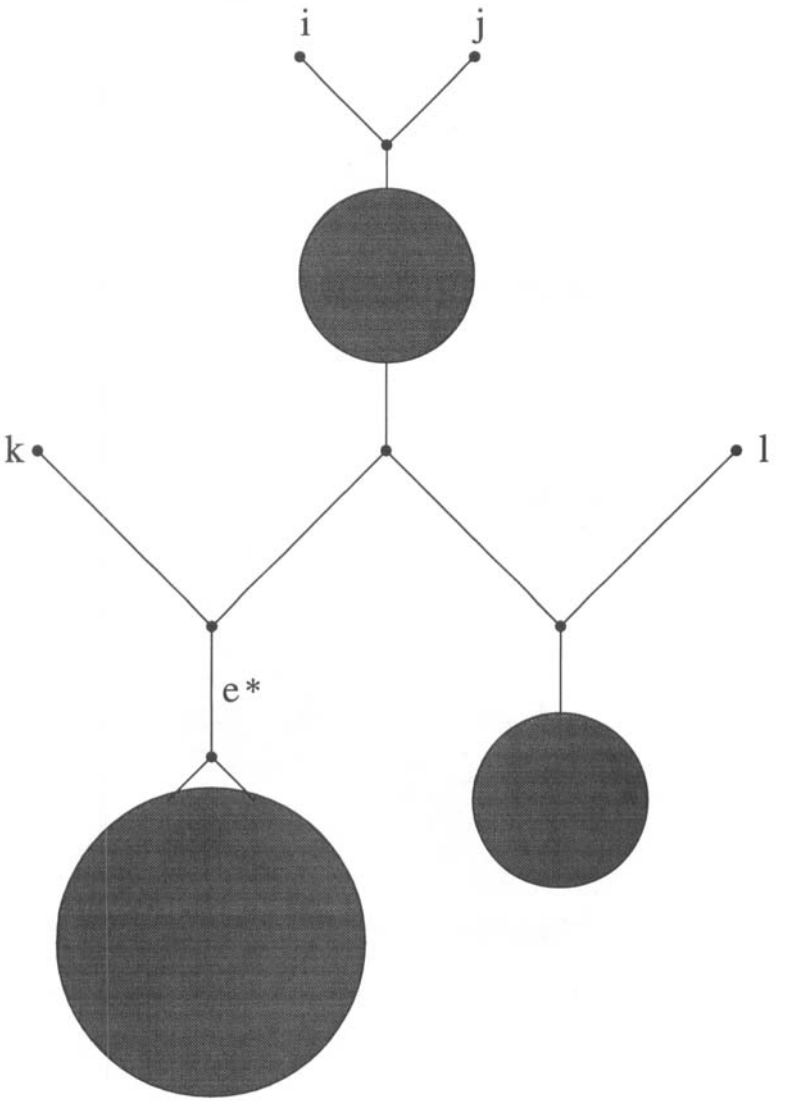


FIGURE 3. An illustration of the choice of neighbors in Lemma 3. The neighbors are chosen in any non-maximal component coming off of $P_{k,l}$, the path between k and l.

3. Suppose $e \in P_{i,j}$. This case is subsumed by case 2 since i and j are symmetric.
4. Suppose $e \in E - P_{i,j} - P_{k,l} - P_{i,k}$. In this case, $\{i, j, k, l\} \subseteq L_k(e)$. Hence, the $w_e(i, j) = -2(|L| - |L_i(e)|) = -2(|L| - |L_k(e)|) = w_e(k, l)$.

Hence, assuming $\{i, j\} \cap \{k, l\} = \emptyset$ and using the differences in weights found in item 2b and item 1:

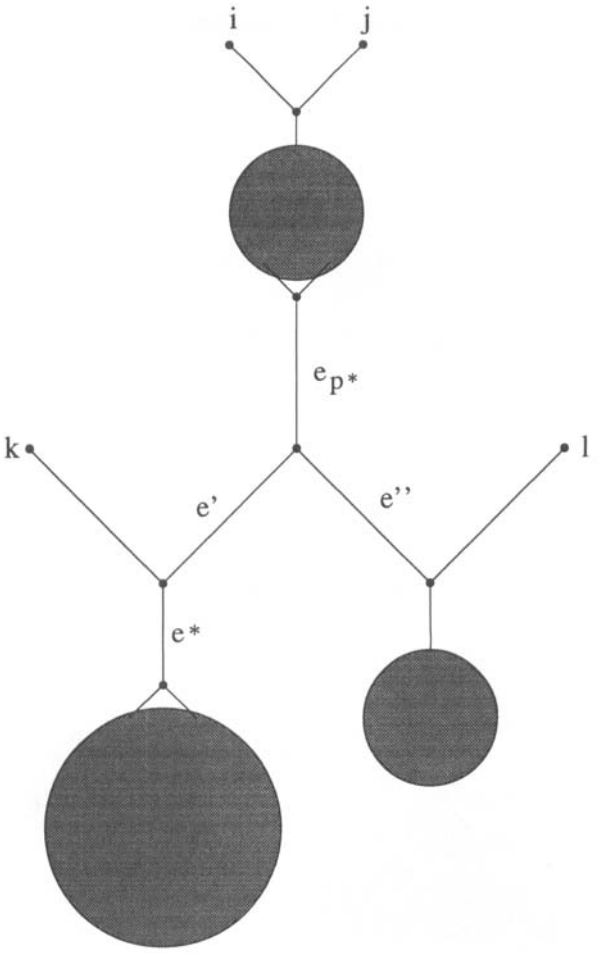


FIGURE 4. Illustration of Case 1 of the proof of Lemma 3.

$$\begin{aligned}
S_{k,l} - S_{i,j} &= \sum_{e \in E} (w_e(k,l) - w_e(i,j))\, l(e) \\
&\geq \sum_{e \in \{e', e'', e'''\}} (w_e(k,l) - w_e(i,j))\, l(e) \\
&= (w_{e'}(k,l) - w_{e'}(i,j)) l(e'') + (w_{e''}(k,l) - w_{e''}(i,j)) l(e'') \\
&\quad + (w_{e'''}(k,l) - w_{e'''}(i,j)) l(e''') \\
&\geq (w_{e'}(k,l) + w_{e''}(k,l) - w_{e'}(i,j) - w_{e''}(i,j) \\
&\quad + w_{e'''}(k,l) - w_{e'''}(i,j)) \min_e l(e'') \\
&\geq (-4 - -2(|L| - 2) + |L| - 4) \min_e l(e) = 3(|L| - 4) \min_e l(e)
\end{aligned}$$

Similarly, assuming $\{i,j\} \cap \{k,l\} \neq \emptyset$ and using the difference in weights found in item 2b:

$$\begin{aligned} S_{k,l} - S_{i,j} &= \sum_{e \in E} (w_e(k,l) - w_e(i,j))\, l(e) \\ &\geq (w_{e'''}(k,l) - w_{e'''}(i,j)) l(e''') \\ &\geq (-2 - -2(|L| - 2)) \min_e l(e) = 2(|L| - 3) \min_e l(e) \end{aligned}$$

Hence, the theorem is proved. □

We now introduce the notion that the algorithm is not given the exact distances, D, of an additive tree but some distances $\hat{D}$ which we will eventually assume are sufficiently near D. As before, S denotes the results of formula (1) applied to distances D but also $\hat{S}$ the same results for distances $\hat{D}$. We wish to show that for any non-neighbors k, l, there are neighbors i, j such that $\hat{S}_{k,l} > \hat{S}_{i,j}$. First, we decompose:

$$\hat{S}_{k,l} - \hat{S}_{i,j} = \hat{S}_{k,l} - S_{k,l} + S_{k,l} - S_{i,j} + S_{i,j} - \hat{S}_{i,j}$$

Lemma 3 bounds the middle pair of terms of the above and so it would be natural to seek a bound for $\left|\hat{S}_{k,l} - S_{k,l}\right|$ for any pair k, l. However, this does not lead to the tightest results and so we instead bound $\hat{S}_{k,l} - S_{k,l} + S_{i,j} - \hat{S}_{i,j}$ which, due to a cancelation of terms, yields a tighter bound:

LEMMA 4. *Let D and $\hat{D}$ denote two distance matrices. We have:*

$$\hat{S}_{k,l} - S_{k,l} + S_{i,j} - \hat{S}_{i,j} \geq -6(|L| - 4) \max_{i,j} \left|D_{ij} - \hat{D}_{ij}\right|$$

when $\{i,j\} \cap \{k,l\} = \emptyset$ and, when $\{i,j\} \cap \{k,l\} \neq \emptyset$,

$$\hat{S}_{k,l} - S_{k,l} + S_{i,j} - \hat{S}_{i,j} \geq -4(|L| - 3) \max_{i,j} \left|D_{ij} - \hat{D}_{ij}\right|$$

PROOF. First note that if $\{i,j\} = \{k,l\}$ then $\left|\hat{S}_{i,j} - S_{i,j} + S_{k,l} - \hat{S}_{k,l}\right| = 0$ and so the result holds in this case. Let $\epsilon_{i,j} = \hat{D}_{ij} - D_{ij}$. We have, from (1):

$$\begin{aligned} &\hat{S}_{i,j} - S_{i,j} + S_{k,l} - \hat{S}_{k,l} \\ &\quad = (|L| - 2)(\epsilon_{i,j} - \epsilon_{k,l}) + \sum_m (\epsilon_{k,m} + \epsilon_{l,m} - \epsilon_{i,m} - \epsilon_{j,m}) \end{aligned} \tag{6}$$

There are two remaining cases:

1. Suppose $|\{i,j\} \cap \{k,l\}| = 1$. Assume without loss of generality that $i = k$. In this case, (6) reduces to:

$$\begin{aligned}
&\hat{S}_{k,l} - S_{k,l} + S_{i,j} - \hat{S}_{i,j} \\
&\quad = (|L| - 2)(\epsilon_{k,l} - \epsilon_{i,j}) + \sum_m (\epsilon_{j,m} - \epsilon_{l,m}) \\
&\quad = (|L| - 3)(\epsilon_{k,l} - \epsilon_{i,j}) + \sum_{m \notin \{j,k,l\}} \epsilon_{j,m} - \sum_{m \notin \{i,j,l\}} \epsilon_{l,m} \\
&\quad \geq -4(|L| - 3) \max_{i,j} \left| D_{ij} - \hat{D}_{ij} \right|
\end{aligned}$$

Hence, the result holds in this case.

2. Otherwise $\{i, j\} \cap \{k, l\} = \emptyset$ and so:

$$\begin{aligned}
&\hat{S}_{k,l} - S_{k,l} + S_{i,j} - \hat{S}_{i,j} \\
&\quad = (|L| - 4)(\epsilon_{k,l} - \epsilon_{i,j}) + \sum_{m \notin \{i,j,k,l\}} (\epsilon_{i,m} + \epsilon_{j,m} - \epsilon_{k,m} - \epsilon_{l,m}) \\
&\quad \geq -6(|L| - 4) \max_{i,j} \left| D_{ij} - \hat{D}_{ij} \right|
\end{aligned}$$

and so the result holds in this case.

□

Now we consider the iterative nature of the algorithm. We consider the behavior of the algorithm when given the approximate distances $\hat{D}$ as input assuming update formula (2) is used. Let i^n and j^n be the taxa chosen as neighbors at the *nth* iteration and u^n the taxon into which they are combined. We let $\hat{D}^n$ denote the distances input into the *nth* iteration. In particular, $\hat{D}^1 = \hat{D}$ and:

$$\hat{D}^{n+1}_{kl} = \begin{cases} \hat{D}^n_{kl} & \text{if } k, l \neq u^n \\ \lambda_{u^n} \hat{D}^n_{il} + (1 - \lambda_{u^n}) \hat{D}^n_{jl} & \text{if } k = u^n \end{cases}$$

Similarly, let $\hat{S}^n$ denote the results of formula (1) evaluated at the distances $\hat{D}^n$. Hence, i^n and j^n are chosen such that $\hat{S}^n_{i^n j^n}$ is minimal. Also, let D^n denote the "actual" distances at the *nth* iteration, when the neighbors which are chosen using $\hat{D}$ are combined. In other words, $D^1 = D$ and:

$$D^{n+1}_{kl} = \begin{cases} D^n_{kl} & \text{if } k, l \neq u^n \\ \lambda_{u^n} D^n_{il} + (1 - \lambda_{u^n}) D^n_{jl} & \text{if } k = u^n \end{cases}$$

It is important to note that we are using the neighbors i^n and j^n chosen by the algorithm using $\hat{D}$ and not D. Finally, we let S^n denote the results of formula (1) evaluated at the distances D^n. We now show that the distances D^n and $\hat{D}^n$ do not grow further apart.

LEMMA 5. *For any n, we have:*

$$\max_{i,j} \left| \hat{D}^n_{ij} - D^n_{ij} \right| \leq \max_{i,j} \left| \hat{D}_{ij} - D_{ij} \right|$$

PROOF. The proof is by induction. The result holds for $n = 1$ by definition.

Now suppose that the result holds for n. If $k, l \neq u^n$, then the distances are unchanged and so $\left|\hat{D}_{kl}^{n+1} - D_{kl}^{n+1}\right| = \left|\hat{D}_{kl}^{n} - D_{kl}^{n}\right|$. Otherwise:

$$\begin{aligned}
\left|\hat{D}_{u^n k}^{n+1} - D_{u^n k}^{n+1}\right| &= \left|\left(\lambda_{u^n}\hat{D}_{i^n k}^{n} + (1-\lambda_{u^n})\hat{D}_{j^n k}^{n}\right) - \left(\lambda_{u^n} D_{i^n k}^{n} + (1-\lambda_{u^n}) D_{j^n k}^{n}\right)\right| \\
&= \left|\lambda_{u^n}\left(\hat{D}_{i^n k}^{n} - D_{i^n k}^{n}\right) + (1-\lambda_{u^n})\left(\hat{D}_{j^n k}^{n} - D_{j^n k}^{n}\right)\right| \\
&\leq \lambda_{u^n}\left|\hat{D}_{i^n k}^{n} - D_{i^n k}^{n}\right| + (1-\lambda_{u^n})\left|\hat{D}_{j^n k}^{n} - D_{j^n k}^{n}\right| \\
&\leq \lambda_{u^n}\max_{i,j}\left|\hat{D}_{ij}^{n} - D_{ij}^{n}\right| + (1-\lambda_{u^n})\max_{i,j}\left|\hat{D}_{ij}^{n} - D_{ij}^{n}\right| \\
&= \max_{i,j}\left|\hat{D}_{ij}^{n} - D_{ij}^{n}\right|
\end{aligned}$$

Hence, the result holds. □

THEOREM 1. *If D corresponds with an additive binary tree T and if:*

$$\max_{i,j}\left|\hat{D}_{ij} - D_{ij}\right| < \frac{\min_e l(e)}{2}$$

then the neighbor-joining algorithm reconstructs T when given $\hat{D}$ as input.

PROOF. First we prove that if D^n corresponds with an additive tree $T^n = (V^n, E^n)$ such that $\min_{e\in E^n} l(e) \geq \min_{e\in E} l(e)$, then $\hat{S}^n$ is minimized at a pair of neighbors in this tree. Note that at the *nth* iteration D^n is a distance matrix on $N - n + 1$ taxa since two taxa are combined at each iteration. Choose a pair of non-neighbors k and l in T^n and let i and j be the pair of neighbors in T^n whose existence is hypothesized in the statement of Lemma 3. Suppose first that $\{k, l\} \cap \{i, j\} = \emptyset$:

$$\begin{aligned}
\hat{S}_{k,l}^{n} - \hat{S}_{i,j}^{n} &= \hat{S}_{k,l}^{n} - S_{k,l}^{n} + S_{i,j}^{n} - \hat{S}_{i,j}^{n} + S_{k,l}^{n} - S_{i,j}^{n} \\
&\geq -6(N-n-3)\max_{k,l\in L}\left|D_{k,l}^{n} - \hat{D}_{k,l}^{n}\right| + 3(N-n-3)\min_{e\in E^n} l(e) \\
&\geq -6(N-n-3)\max_{k,l\in L}\left|D_{k,l} - \hat{D}_{k,l}\right| + 3(N-n-3)\min_{e\in E} l(e) \\
&> 0
\end{aligned}$$

where we have used Lemma 3 for the first inequality, Lemma 4 and the assumption that $\min_{e\in E^n} l(e) \geq \min_{e\in E} l(e)$ for the second and Lemma 5 for the third. Similarly, if $\{k, l\} \cap \{i, j\} \neq \emptyset$, we have:

$$\begin{aligned}
\hat{S}_{k,l}^{n} - \hat{S}_{i,j}^{n} &= \hat{S}_{k,l}^{n} - S_{k,l}^{n} + S_{i,j}^{n} - \hat{S}_{i,j}^{n} + S_{k,l}^{n} - S_{i,j}^{n} \\
&\geq -4(N-n-2)\max_{k,l\in L}\left|D_{k,l}^{n} - \hat{D}_{k,l}^{n}\right| + 2(N-n-2)\min_{e\in E^n} l(e) \\
&\geq -4(N-n-2)\max_{k,l\in L}\left|D_{k,l} - \hat{D}_{k,l}\right| + 2(N-n-2)\min_{e\in E} l(e) \\
&> 0
\end{aligned}$$

Hence, if D^n corresponds with an additive tree and $\min_{e\in E^n} l(e) \geq \min_{e\in E} l(e)$ then $\hat{S}^n$ is minimized at a pair of neighbors in D^n.

Now we show by induction that D^n corresponds with an additive tree and that $\min_{e\in E^n} l(e) \geq \min_{e\in E} l(e)$. The base case, $n = 1$, follows by assumption of the theorem. Now suppose that the D^n corresponds with an additive tree and that $\min_{e\in E^n} l(e) \geq \min_{e\in E} l(e)$. By the previous paragraph, the $\hat{S}^n$ is minimized at a pair of neighbors i^n and j^n in D^n. Hence, by Lemma 1, D^{n+1} is additive. Also by Lemma 1, $\min_{e\in E^{n+1}} l(e) \geq \min_{e\in E^n} l(e)$. This is because for all $e \in E^{n+1}$, either $e \in E^n$ or $e = \{u, u''\}$. In either case $l(e) \geq \min_{e\in E^n} l(e)$ since in the later case $l(\{u, u''\}) \geq l(\{u', u''\})$ and $\{u', u''\} \in E^n$. Hence, by the induction hypothesis, $\min_{e\in E^{n+1}} l(e) \geq \min_{e\in E} l(e)$.

Hence, putting together the results from the previous two paragraphs, the neighbor-joining algorithm chooses a pair of neighbors at every iteration. Now let T^n denote the tree corresponding to D^n and let L^n denote the set of leaves of T^n. For any leaf $u \in L^n$, define the representatives of u, written L_u, be the set of leaves of T which have been combined to form u, that is, L_u is defined by the following inductive definition:

$$L_u = \begin{cases} L_i \cup L_j & \text{if } u = u^n \text{ for some } n \\ \{u\} & \text{otherwise} \end{cases}$$

We will say that a set of leaves $L' \subseteq L$ is monophyletic if there is an edge $e(L')$ such that L' and $L - L'$ are contained in different components of $T - e$. For any monophyletic set of leaves L', let $T(L')$ denote the component of $T - e$ containing L'. Note that $T(L')$ is a rooted binary tree. Finally, let $T^{n\prime}$ denote the tree formed from T^n by adding the tree $T(L_u)$ with its root at u for each leaf $u \in L^n$.

We will now demonstrate by induction that L_u is monophyletic for every u and that if $T^{n\prime}$ has the same topology as T. The base case is clear since $L_u = \{u\}$ and $T^{1\prime} = T$. Now we assume that the result holds for n and demonstrate that it is true for $n + 1$. By the induction hypothesis, $L(i^n)$ and $L(j^n)$ are monophyletic. From the previous paragraph, i^n and j^n are neighbors in T^n. As in Lemma 1, let $u^{n\prime}$ denote the internal vertex adjacent to i^n and j^n and $u^{n\prime\prime}$ the vertex adjacent to $u^{n\prime}$ but different from i^n and j^n. Since the edge $\{u^{n\prime}, u^{n\prime\prime}\}$ separates $\{i^n, j^n\}$ from $L^n - \{i^n, j^n\}$ and since $T^{n\prime}$ has the same topology as T^n, it must be that $\{u^{n\prime}, u^{n\prime\prime}\}$ separates $L(i^n) \cup L(j^n)$ from $L - (L(i^n) \cup L(j^n))$ and so $L(i^n) \cup L(j^n)$ is monophyletic. Now, the topology of T^{n+1} is that of T^n with i^n and j^n removed by Lemma 1. Hence, $T^{n+1\prime}$ has the same topology as $T^{n\prime}$ since $T(u^n)$ is formed from $T(i^n)$ and $T(j^n)$ by joining their roots to its root.

Hence, at each step, the algorithm finds a new monophyletic group of size at least 2, namely, $L(u^n)$. There are exactly $n - 3$ such groups since these groups correspond with internal edges (edges not adjacent to leaves). Since the algorithm runs for $n - 3$ steps (there is no need to proceed beyond the point at which there are 3 remaining taxa), it finds all such groups. As is well known, the tree topology is completed determined by the monophyletic groups since these groups correspond with split[**BG91**]. □

3. Acknowledgements

I'd like to thank Tandy Warnow for suggesting this problem and for useful discussions related to it and Olivier Gascuel for many suggested corrections to the manuscript.

References

[ABF+96] Richa Agarwala, Vineet Bafna, Martin Farach, Babu O. Narayanan, Mike Paterson, and Mikkel Thorup, *On the approximability of numerical taxonomy*, Proceedings of the Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, 1996, pp. 365–372.

[BD86] Hans-Jürgen Bandelt and Andreas Dress, *Reconstructing the shape of a tree from observed dissimilarity data*, Advances in Applied Mathematics **7** (1986), 309–343.

[BG91] Jean-Pierre Barthélemy and Alain Guénoche, *Trees and proximity representations*, John Wiley & Sons, 1991.

[Cav78] James A. Cavender, *Taxonomy with confidence*, Mathematical Biosciences **40** (1978), 271–280.

[Far73] James S. Farris, *A probability model for inferring evolutionary trees*, Systematic Zoology **22** (1973), 250–256.

[FK96] Martin Farach and Sampath Kannan, *Efficient algorithms for inverting evolution*, Proceedings of the Twenty-Eigth Annual ACM Symposium on the Theory of Computing, 1996, pp. 230–235.

[Gas94] Olivier Gascuel, *A note on sattah and tversky's, saitou and nei's, and studier and keppler's algorithms for inferring phylogenies from evolutionary distances*, Mol. Biol. Evol. **11** (1994), no. 6, 961–963.

[GS92] G. R. Grimmett and D. R. Stirzaker, *Probability and random processes*, Oxford University Press, 1992.

[SK88] James A. Studier and Karl J. Keppler, *A note on the neighbor-joining algorithm of saitou and nei*, Mol. Biol. Evol. **5** (1988), no. 6, 729–731.

[SN87] N. Saitou and M. Nei, *The neighbor-joining method: a new method for reconstructing phylogenetic trees*, Mol. Biol. Evol. **4** (1987), 406–425.

[SOWH96] D. L. Swofford, G. J. Olsen, P. J. Waddell, and D. M. Hillis, *Phylogenetic inference*, Molecular Systematics (D.M. Hillis, C. Moritz, and B.K. Mable, eds.), Sinauer Associates, Inc., 1996, pp. 407–514.

[ST77] Shmuel Sattath and Amos Tversky, *Additive similarity trees*, Psychometrika **42** (1977), no. 3, 319–345.

1315 Blockley Hall, 418 Guardian Drive, Philadelphia, PA 19104
E-mail address: atteson@dnamite.humgen.upenn.edu

This page intentionally left blank

https://doi.org/10.1090/dimacs/037/09

DIMACS Series in Discrete Mathematics
and Theoretical Computer Science
Volume **37**, 1997

Concerning the NJ Algorithm and Its Unweighted Version, UNJ

Olivier Gascuel

ABSTRACT. In this paper we will present UNJ, the unweighted version of the NJ algorithm (Saitou and Nei 1987; Studier and Keppler 1988). We will demonstrate that UNJ is well suited when the data are of the $(\delta_{ij}) = (d_{ij} + \varepsilon_{ij})$ type, where (d_{ij}) is a tree distance and the ε_{ij} are independent and identically distributed noise variables. Simulations confirm this theory. On a more general level, we study the three main components of the agglomerative approach, applied to the reconstruction of tree distances.

(i) We will demonstrate that the selection criterion for the pair to be agglomerated, used by NJ and UNJ, retains its meaning whatever the variances and covariances of the δ_{ij} estimates. We will also provide a new proof of the correction of this criterion, based on its interpretation in terms of acentrality index proposed by Mirkin (1996).

(ii) Using the results of Vach (1989), of which we will provide a simple new demonstration, we propose an analytical formula which enables the correct least-squares estimation of edge lengths in $O(n^2)$ time, where n is the number of objects.

(iii) We will provide a class of admissible reduction formulae which guarantee the finding of the true tree with additive data. We propose to choose, among these formulae, the minimum variance reduction, so that at each step we use estimates which are as reliable as possible in choosing the pair to be agglomerated. We will present the general solution, and apply it to the particular data model retained here.

1. Introduction

Let $\mathbf{D} = (d_{ij})$ be a tree distance over n objects; $\mathbf{T}$ the unique valued tree admitting the representation $\mathbf{D}$, also referred to as the true tree; $\mathbf{\Delta} = (\delta_{ij})$ a dissimilarity matrix of which each element δ_{ij} is an estimate or a measure, generally imperfect, of the distance d_{ij}. In this paper, we propose to find $\mathbf{T}$ from the observation $\mathbf{\Delta}$. In other words, we try to construct a valued tree $\hat{\mathbf{T}}$, associated with the tree distance $\hat{\mathbf{D}} = (\hat{d}_{ij})$, which should be as close as possible to $\mathbf{T}$. Of course, there are several ways of defining the proximity between trees. In this paper, we will focus on the structure of the trees $\mathbf{T}$ and $\hat{\mathbf{T}}$, rather than on the length of their edges. This problem is encountered in domains where one tries to construct an inheritance phenomenon as, for example, the history of manuscripts in Archaeology (Buneman 1971), or the evolution of the species in Biology (Swofford *et al.* 1996). The tree $\mathbf{T}$ represents the history, the distances d_{ij} represent the divergence times

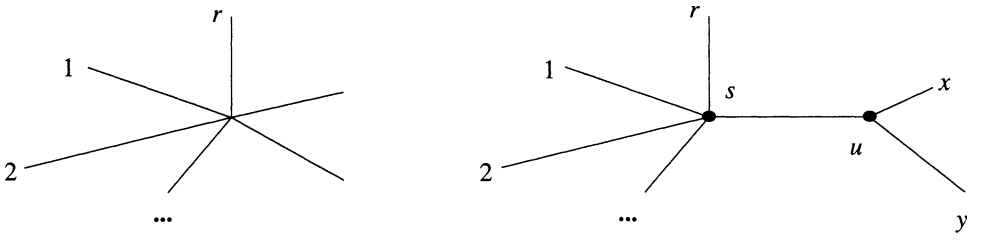
1991 *Mathematics Subject Classification.* 62H30, 62P10, 62P15.

between these objects or these species, and the dissimilarities δ_{ij} are estimates of these divergence times.

A classical approach consists in following the least-squares criterion in constructing the positively-valued tree which best represents $\mathbf{D}$ according to this criterion (Cunningham 1978; De Soete 1983; Roux 1988; Gascuel and Levy 1996). Although simulation results are good (Kuhner and Felsenstein 1994; Gascuel and Levy 1996; Kumar 1996), this approach is not entirely satisfactory, since, to the best of our knowledge, very few results establish a link between the structure of the tree $\hat{\mathbf{T}}$ thus inferred and the structure of the true tree $\mathbf{T}$. This lacuna is partially filled by another approach, which is widely used in the domain of Evolution (Kidd and Sgaramella-Zonta 1971; Saitou and Nei 1987), and which is called the minimum evolution principle (ME). This principle consists in seeking among all the possible tree structures, that which leads to the "shortest" valued tree. The length of a valued tree is the sum of the lengths (valuations) of its edges, and within the ME principle, these are estimated using the least-squares criterion, without the positivity constraint. The structure of $\hat{\mathbf{T}}$ being fixed, the length of the edges are thus obtained by minimizing the Euclidean distance between $\hat{\mathbf{D}}$ and $\mathbf{\Delta}$. This minimization problem has a unique solution (described below, Section 2), and the length associated with a tree structure is thus well defined. Rzhetsky and Nei (1993) provide a thorough justification of the ME principle. They demonstrate that if the estimates (δ_{ij}) are unbiased, *i.e.*, $E(\delta_{ij}) = (d_{ij})$, then the structure of the true tree $\mathbf{T}$ has among all the possible tree structures, the shortest expected length. This property justifies the ME principle which, in order to find the true tree, seeks the shortest tree. To some extent, the probability of finding the true tree is thus maximized.

Saitou and Nei (1987) proposed an agglomerative algorithm, called NJ (Neighbor-Joining) which is based on the ME principle. A second version of this algorithm was proposed by Studier and Keppler (1988). It is equivalent to the original version (Gascuel 1994), but simpler. NJ may be described as follows. At each step, one considers r nodes, each representing either an object or a group of objects agglomerated during the previous steps, on which the tree $\hat{\mathbf{T}}$ remains to be constructed (Figure 1a). A criterion, denoted S, enables one to choose, among the nodes, the pair to be agglomerated. This criterion is, in a way, equal to the least-squares length estimate of the tree represented in Figure 1b. Once the pair, $\{x, y\}$ say, is chosen, we create the node u (Figure 1b) and we determine the length of the edges (x, u) and (y, u). Then we replace the nodes x and y by the node $\mathbf{u}$ in the dissimilarity matrix, by setting $\delta_{ui} = (\delta_{xi} + \delta_{yi})/2$ for each i different from x and y. The process is iterated until r is equal to 3. Finally, we join the 3 remaining objects to a central node and we compute the length of the last 3 edges. This algorithm thus follows the classical scheme of bottom-up hierarchical clustering, already used for tree distances by Sattath and Tversky (1977). Inside this scheme, NJ is defined by the three basic components: (i) the selection criterion for the pair to be agglomerated; (ii) the calculation method for the edge lengths; (iii) the formula enabling the reduction of the dissimilarity matrix.

Although the NJ algorithm is widely used and has yielded satisfactory simulation results (Nei 1991; Gascuel and Levy 1996), certain questions remain concerning each of its components.



(a) (b)

FIGURE 1. (**a**) star tree representing absence of structure; (**b**) tree in which the pair $\{x, y\}$ has been agglomerated.

(i) When applied to a tree distance $\mathbf{D}$, the selection criterion is expected to systematically designate a true pair of neighbors of the corresponding tree. Saitou and Nei's (1987) proof concerning the correctness of the criterion S has been contested by Studier and Keppler (1988) who provide a new proof, which in turn has been contested by Mirkin (1996). Charleston *et al.* (1993) present an interesting study concerning this criterion, but they do not provide a complete proof of its correctness either.

(ii) The manner in which the edge lengths are estimated by NJ is inexact, in terms of the least-squares, when the agglomerated nodes represent not individual objects but rather groups of objects. This deficiency has inspired several authors (Sattath and Tversky 1977; Brölsch 1983; Vach 1989; Vach and Degens 1991; Rzhetsky and Nei 1993) to propose other formulae which are exact, but which are difficult to integrate into the agglomerative procedure.

(iii) Finally, one may question the true meaning of the NJ reduction formula which, systematically, gives identical importance to nodes x and y, even when one corresponds to a group comprising several objects and the other corresponds to a single object. In terms of hierarchical classification, such a reduction is said to be "weighted", meaning that the objects will not have the same influence depending on whether they belong to a large group or are isolated.

This paper will provide answers to each of these questions. In order to make the context precise, we place a model on the data making the hypothesis that the estimates (δ_{ij}) are unbiased, mutually independent and have the same variance. A slightly more specific version of this hypothesis consists in setting $\delta_{ij} = d_{ij} + \varepsilon_{ij}$ for every i and j, where the noise variables ε_{ij} are i.i.d. (independent and identically distributed) and of null expectation. This hypothesis is commonly applied when the estimates are a result of real observations with measurement errors. Within the scope of this model, we will demonstrate that it is meaningful to use an unweighted approach which allocates the same level of importance to each of the initial objects. Furthermore, within this model it is justified to use the "ordinary" least-squares criterion (to estimate the length of the edges) as opposed to the "generalized" least-squares criterion which takes into account the variances and covariances of the estimates (Searl 1971; Bulmer 1991). Taking into consideration all of the above factors leads to the unweighted version of NJ, which will be called UNJ in what follows.

This paper is organized as follows. Firstly, we will provide a certain number of definitions, notations, and previous results (Section 2). Then, we will describe

the UNJ algorithm and its main properties (Section 3). This algorithm follows the same agglomerative scheme as NJ and is defined by the three components described above which will be studied sequentially. Section 4 demonstrates that the selection criterion used by UNJ (identical to that of NJ) retains its meaning whatever the variances and covariances of the δ_{ij} estimates, and that it is correct, in that it always selects a true pair of neighbors when the data are additive. In Section 5, we demonstrate that the formula used by UNJ is correct in the least-squares sense. In order to establish this property, we use a fundamental yet relatively unknown result of Vach (1989), for which we provide a new, very simple proof. In Section 6, we show that the unweighted reduction used by UNJ is coherent with our data model. Section 7 compares performances of NJ and UNJ on simulated data, while Section 8 is devoted to discussion.

2. Preamble

The $\mathbf{\Delta} = (\delta_{ij})$ dissimilarity is over the set E of n objects and we denote $E = \{1, 2, \ldots, n\}$. The trees considered here have n leaves which are labeled with each of the objects of E. Throughout this paper, a distinction will be made between a valued tree and its structure. Any valued tree will be denoted $\mathbf{S}$, while its structure will be denoted $\dot{\mathbf{S}}$. The structure of the $\mathbf{S}$ tree is defined by the set of bipartitions of E corresponding to each of its edges (by removing an edge from $\mathbf{S}$, we cut E into two disjoint subsets). Each of these bipartitions constitutes a pair $\{X, \overline{X}\}$ where X may be viewed in two different ways: as a subset of E (thus we have $\overline{X} = E - X$); or as a rooted subtree of $\mathbf{S}$ whose root is situated at the extremity of the edge in question ($\overline{X}$ is then the subtree situated on " the other side extremity" of the edge). Depending on the context, we will use one or other of these notions. We will discriminate between trivial bipartitions which separate a unique object E from all the remaining objects and which are always contained in the trees studied herein, and non-trivial bipartitions which separate subsets containing at least two objects. The cardinality of X (or equivalently the number of leaves of the tree X) will be denoted n_X. The root of a tree will be indicated by the corresponding lower case letter, *i.e.*, x will be the root of X, and n_X will sometimes be denoted n_x depending on the context.

The trees inferred by the agglomerative methods such as NJ are binary, *i.e.*, all their internal nodes are of degree 3. Every tree distance may be represented by a tree of this type, provided the zero-valued edges are accepted, so that this characteristic is not restrictive. Nonetheless, such a representation is not always unique, since a 4-degree node may, for example, be represented in three different ways by two 3-degree nodes separated by a zero-valued edge. In order to avoid this difficulty, which slows down the demonstrations, we suppose in what follows that the true tree $\mathbf{T}$ is itself a binary tree and contains only edges which are strictly positive. This restriction has little practical effect, given that every tree distance may be approached as closely as desired by a tree respecting this condition.

One of our objectives here is to demonstrate a simple expression of the least-squares estimation of the edge lengths of a tree whose structure is fixed. We will refer only to estimation without the positivity constraint, which is justified in the case of the minimum evolution principle. A tree which has negative valuations does not define a distance, but an "unsigned tree dissimilarity" (Bandelt and Steel 1995) in which the dissimilarity between two nodes is simply the sum of the valuations

of the path linking these nodes. Given a Δ matrix, the least-squares estimation associates with the tree structure $\dot{S}$, a valued tree which we will refer to as the "adjusted" tree of $\dot{S}$, and which we will denote as S for simplification purposes, Δ being implicit. This adjusted tree is itself associated with an unsigned tree dissimilarity, also simply denoted $S = (s_{ij})$. The matrix expression of S as a function of $\dot{S}$ and of Δ is well known (Sattath and Tversky 1977; Barthlemy and Gunoche 1991). Let us call q the number of edges of $\dot{S}$, and let us suppose that an order has been chosen for the edges, which need not be stated explicitly for our purpose here. Thus we may represent all the edge lengths by a vector $\mathbf{B} = (b_1, b_2, \ldots, b_q)$. S may also be represented by a vector, and we have

$$\mathbf{S}^t = \mathbf{A}\mathbf{B}^t \tag{1}$$

where $\mathbf{A}$ is a 0-1 matrix $\left(n(n-1)/2\right) \times q$ which represents $\dot{S}$. This matrix is defined in the following manner:

$$\mathbf{A}_{(ij)k} = \begin{cases} 1 & \text{if the } k^{\text{th}} \text{ edge (or bipartition) of } \dot{S} \text{ separates } i \text{ and } j, \\ 0 & \text{otherwise,} \end{cases}$$

where (ij) represents the rank of s_{ij} in the vector representation of S. In this framework, S is the projection of Δ on the sub-space generated by the bipartitions of $\dot{S}$, and we get

$$\mathbf{S}^t = \mathbf{A}(\mathbf{A}^t\mathbf{A})^{-1}\mathbf{A}^t\Delta^t. \tag{2}$$

The drawback in this expression is that it requires the computation of the matrix product $\mathbf{A}^t\mathbf{A}$, whose complexity in time is $O(q^2n^2)$. Since the trees under consideration are binary, we get $q = 2n-3$, and the computation is thus in $O(n^4)$. However, as we will see below (Section 5), this expression (2) is fundamental to analytical formulae which enable a more rapid computation of edge lengths, in $O(n^2)$.

Before concluding this section, we introduce some notation. Let S be an adjusted tree, and $\{X, \overline{X}\}$ and $\{Y, \overline{Y}\}$ two bipartitions of $\dot{S}$. When $X \cap Y = \emptyset$, we set by extension

$$\delta_{XY} = \sum_{i \in X, j \in Y} \delta_{ij} \text{ and } \overline{\delta_{XY}} = \frac{1}{n_X n_Y} \sum_{i \in X, j \in Y} \delta_{ij},$$

as well as

$$s_{XY} = \sum_{i \in X, j \in Y} s_{ij} \text{ and } \overline{s_{XY}} = \frac{1}{n_X n_Y} \sum_{i \in X, j \in Y} s_{ij}.$$

We will also refer to the "flow" in S of a rooted subtree X. This quantity, denoted f_X, is the sum of the lengths of the paths between each leaf of X and the root of X. Let x be this root, we get

$$f_X = \sum_{i \in X} s_{ix} \text{ and } \overline{f_X} = \frac{1}{n_X} f_X.$$

3. The UNJ algorithm

The UNJ algorithm is summarized in Figure 2. At each step, it uses and then reduces a running matrix of size $r \times r$ that we have denoted $\Lambda = (\lambda_{ij})$, in order to avoid confusion with the initial data matrix. This matrix is of course initialized with value Δ.

UNJ is defined by the following three formulae:

selection criterion

$$Q_{xy} = R_x + R_y - (r-2)\lambda_{xy}, \quad \text{with} \quad R_z = \sum_{i=1}^{r} \lambda_{zi}, \tag{3}$$

estimation formula

$$\hat{d}_{xu} = \frac{1}{2}\lambda_{xy} + \frac{1}{2(n-n_u)} \sum_{\substack{i=1 \\ \neq x,y}}^{r} n_i(\lambda_{xi} - \lambda_{yi}), \tag{4}$$

$\hat{d}_{yu}$ obtained by symmetry, and

reduction formula

$$\lambda_{ui} = w_x\lambda_{xi} + w_y\lambda_{yi} - w_x\hat{d}_{xu} - w_y\hat{d}_{yu}, \quad \text{where} \quad w_x = \frac{n_x}{n_u} \quad \text{and} \quad w_y = \frac{n_y}{n_u}. \tag{5}$$

Initialize the running matrix: $\Lambda = (\lambda_{ij}) \leftarrow (\delta_{ij})$;
Initialize the number of remaining nodes: $r \leftarrow n$;
Initialize the numbers of objects per node: $n_i \leftarrow 1$, $i \in \{1, \ldots, n\}$;
While the number of nodes r is greater than 3:
{ Compute the sums R_i, $i \in \{1, \ldots, r\}$; **(a)**
Find the pair $\{x, y\}$ to be agglomerated by maximizing Q_{xy} (3); **(b)**
Create the node u, and set: $n_u \leftarrow n_x + n_y$;
Estimate the lengths of edges (x, u) and (y, u) using (4); **(c)**
Reduce the running matrix Λ using (5); **(d)**
Decrease the number of nodes: $r \leftarrow r - 1$ };
Create a central node, and compute the last three edge-lengths using (4);
Output the tree found.

FIGURE 2. The UNJ algorithm.

These three formulae may be easily compared with those proposed by Studier and Keppler (1988) in their simplified version of NJ. The selection criterion is the same. The estimation formula of UNJ is an unweighted version of that of NJ. Indeed, the latter may be expressed as

$$\hat{d}_{xu} = \frac{1}{2}\lambda_{xy} + \frac{1}{2(r-2)} \sum_{\substack{i=1 \\ \neq x,y}}^{r} (\lambda_{xi} - \lambda_{yi}),$$

and is obtained from (4) by setting $n_i = 1$, $\forall\, i \neq x, y$, and by replacing $(n - n_u)$ by $\sum_{i \neq x,y} n_i$ which is then equal to $(r-2)$. Similarly, the NJ reduction formula is obtained from (5) by setting $n_x = n_y = 1$. UNJ, as described here, thus appears as the unweighted version of the NJ algorithm proposed by Studier and Keppler (1988). Its properties are as follows.

PROPERTY 1. *The complexity in time of UNJ is* $O(n^3)$.

PROPERTY 2. *The estimation formula (4), combined with the reduction (5), is optimal, in that whatever the structure of the inferred tree, the edge lengths obtained are identical to those obtained with the matrix solution (2). Furthermore, this*

property may be exploited within an algorithm in $O(n^2)$, allowing the least-squares estimation of edge lengths of any fixed structure binary tree.

PROPERTY 3. *Given our hypothesis on the δ_{ij} estimates, the reduction (5) is optimal in that it minimizes the part of the variance of the running matrix (λ_{ij}) which influences the choice of the structure of $\hat{\mathbf{T}}$. In other words, at each step, it yields estimators which are as reliable as possible in choosing the pair to be agglomerated.*

PROPERTY 4. *When data are additive, UNJ finds the true tree* $\mathbf{T}$.

Property 1 is immediate. In fact, the algorithm carries out $n - 3$ steps, and during each step the most costly operations correspond to lines (a) and (b) which are both in $O(r^2)$. The complexity of UNJ is thus the same as that of NJ.

The first part of Property 2 will be shown in Section 5, while the second part is simple. Indeed, the UNJ algorithm may be transformed into an estimation algorithm of edge lengths of a binary tree with a fixed structure, whose complexity is in $O(n^2)$. To do this, one simply replaces lines (a) and (b) by a tree traversing algorithm which finds in $O(r)$ a pair of neighbors of the tree in question; the most costly remaining lines are now (c) and (d), both being in $O(r)$. This $O(n^2)$ complexity is optimal, since it is linear in the size of the data. An analogous result may be obtained using one (5.3) of the formulae proposed by Vach and Degens (1991). It would appear however that this type of result is relatively unknown, and that users generally opt for the matrix solution, in $O(n^4)$, or for the algorithm of Rzhetsky and Nei (1993) which is in $O(n^3)$. We would like to point out however that formula (4) and this $O(n^2)$ algorithm only apply to binary trees. For non-binary trees, we have to use a more general formula (+) proposed by Vach (1989), and the corresponding algorithm remains to be studied.

Property 3 seems natural since the notions of unweighted mean, of i.i.d. variables and of minimum variance estimator are fundamentally linked. However, this type of argument based on a data model is rarely reviewed in the literature. For example, it does not figure in the comparison between UPGMA and WPGMA proposed by Sneath and Sokal (1973). This property 3 will be made precise and proved in Section 6.

Property 4 is special in that it requires three sub-properties in order to be demonstrated, relating to formulae (3), (4) and (5). Under the hypothesis that the data are additive, *i.e.* $\mathbf{\Delta} = \mathbf{D}$, and that the tree $\mathbf{T}$ representing $\mathbf{D}$ is a binary tree comprising only strictly positive valuations, we will show that:

PROPERTY 5. *Criterion (3) always selects a true pair of neighbors in the tree* $\mathbf{T}$, i.e., *a pair $\{x, y\}$ of leaves linked by a path containing only one interior node, denoted u.*

PROPERTY 6. *The lengths of the edges (x, u) and (y, u) resulting from (4) are identical to those of the corresponding edges in* $\mathbf{T}$.

PROPERTY 7. *Reduction (5) applied to a true neighbor pair $\{x, y\}$ transforms* $\mathbf{D}$ *into an additive matrix* $\mathbf{D}'$ *which is represented by the subtree of* $\mathbf{T}$ *obtained by deleting the nodes x and y and the corresponding edges.*

The combination of these three properties enables us to obtain the result (3.4) by induction, each step of the algorithm consisting in reconstructing correctly (from

the point of view of structure and branch length) two neighboring edges of **T**. At the end, only three nodes remain, and hence only one possible structure, and the computation of the lengths is correct due to Property 6.

Property 5 is shown in Section 4, which fills the lacuna indicated by Mirkin (1996) as outlined in our Introduction. Property 6 is an immediate consequence of Property 2, given that the matrix solution (2) is obviously exact in the case of additive data. Finally, we will show, in Section 6, a generalization of the Property 7 which enables the correctness proof of an entire class of variants of NJ, including among others, NJ itself, UNJ and BIONJ (Gascuel 1996).

4. Concerning the Q selection criterion

UNJ retains the same criterion as NJ to select the node pair to be agglomerated. This criterion is open to several interpretations and several expressions which we will recall briefly (Subsection 4.1). Next (Subsection 4.2), we will present, in greater detail, the interpretation proposed by Mirkin (1996) and we will show how this interpretation has the advantage of retaining its full meaning when we drop the hypothesis that the δ_{ij} estimates are mutually independent and that they have the same variance. Finally (Subsection 4.3), we show that this interpretation leads to a simple proof of the correction of the Q criterion (Property 5).

4.1. The different expressions and interpretations of the Q criterion. At each step, we consider the dissimilarity matrix (λ_{ij}) where i and j represent objects or object clusters already agglomerated during the previous steps. Saitou and Nei (1987), at each step, assimilate these indices with unique objects, and they choose the pair $\{x, y\}$ which minimizes the least-squares length estimate of the tree represented in Figure 1b. The criterion thus defined is denoted S_{xy}, and is expressed

$$S_{xy} = \frac{1}{2}\lambda_{xy} + \frac{1}{2(r-2)} \sum_{\substack{i=1 \\ \neq x,y}}^{r} (\lambda_{1i} + \lambda_{2i}) + \frac{1}{r-2} \sum_{\substack{i=1 \\ \neq x,y}}^{r} \sum_{\substack{j=i \\ \neq x,y}}^{r} \lambda_{ij}. \tag{6}$$

Studier and Keppler (1988) propose replacing the S criterion by the Q criterion defined above (3), without attaching any specific interpretation to the latter. This new expression has the advantage of leading to a complexity in $O(n^3)$; as shown above (Property 1). Furthermore, it is equivalent to the original expression (6), criteria S and Q being linked by a negative slope linear expression (Gascuel 1994). Vach and Degens (1991) indicate that minimizing S (or maximizing Q) is equivalent to maximizing the least-squares length estimate of the edge linking u, the root of the cluster in formation, and s, the center of the star (Figure 1b). We have also shown (Gascuel 1994) that S could be interpreted as a continuous version of the neighborliness criterion proposed by Sattath and Tversky (1977).

The tree 1b does not represent the structure of the true tree. By minimizing at each step its estimated length, one tends to find a short tree, which justifies the use of the S criterion within a greedy algorithm which follows the ME principle. However, the real reason why we systematically obtain the true tree (*i.e.*, the shortest tree) when the data are additive is, to a great extent, inexplicable. The same applies to the other interpretations proposed, which thus seem unsatisfactory. Moreover, the interpretations based on the length of the tree (Saitou and Nei 1987) or of the internal edge (Vach and Degens 1991) lead to a form of contradiction.

Indeed, formula (6), used in estimating the length of tree 1b, is only correct at the first step, when each index designates a unique object. During the following steps, some indices represent object clusters, and this formula becomes approximate. The same applies to the length of the edge (u, s), and Vach and Degens (1991) use an exact formula which is no longer linked, in a linear manner, to S. An optimal formula also exists which gives the least-squares length estimate of tree 1b at each step of the algorithm (available on demand). These optimal formulae are a priori more satisfying than the previous ones, if we accept the proposed interpretations. However, we observe with examples, that neither guarantees the finding of the true tree when the data are additive.

4.2. Interpreting Q in acentrality terms. Mirkin (1996) proposes another interpretation of Q, in acentrality terms. Let us consider for the moment that Q is applied to the tree distance $\mathbf{D}$. Now $Q'_{xy} = Q_{xy}/2$ may be expressed as

$$Q'_{xy} = d_{xy} + \sum_{\substack{i=1 \\ \neq x,y}}^{r} \frac{1}{2}(d_{xi} + d_{yi} - d_{xy}). \tag{7}$$

It is easy to see that the expression inside the sum, *i.e.*, $(d_{xi} + d_{yi} - d_{xy})$, is equal to the length of the path (i, u), where u (Figure 3a) is the intersection of the paths (x, y), (x, i), and (y, i). In other words, this expression measures the acentrality of the path (x, y) for the node i, and Q' is equal to the sum of all these measures, to which d_{xy} is added, which expresses the acentrality of nodes x and y themselves. Q'_{xy} is thus an acentrality measurement of the pair $\{x, y\}$.

Let us consider the examples shown in Figures 3b and 3c. In the first case, we examine the pair of neighbors $\{i, j\}$, while in the second case, we examine the pair $\{i, k\}$. In the case of $\{i, j\}$, Q' is equal to the sum $(d_{ij} + d_{tk} + d_{tl} + d_{tm})$ in which each external edge is counted once, while the edge lengths d_{tu} and d_{uv} are counted, respectively, thrice and twice. In the case of $\{i, k\}$, Q' is equal to the sum $(d_{ik} + d_{tj} + d_{ul} + d_{um})$ in which the external edges are always counted once, but the edge lengths d_{tu} and d_{uv} are now counted, respectively, once and twice. It is evident from this example that when applied to a pair of neighbors, the criterion counts the internal edges several times, whereas when applied to remote nodes in the tree, the criterion tends to count certain edges le ss often. By applying the criterion to the pair $\{i, \ell\}$, or to any pair situated at either extremity of the tree, each edge is only counted once.

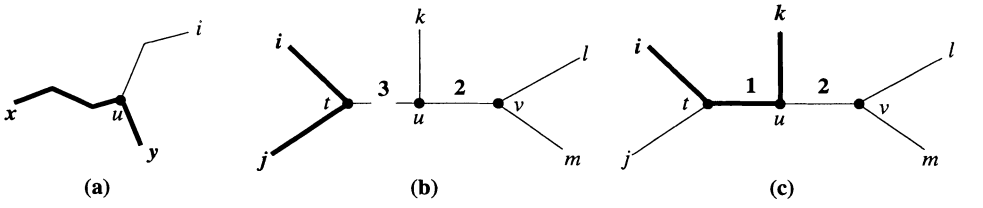


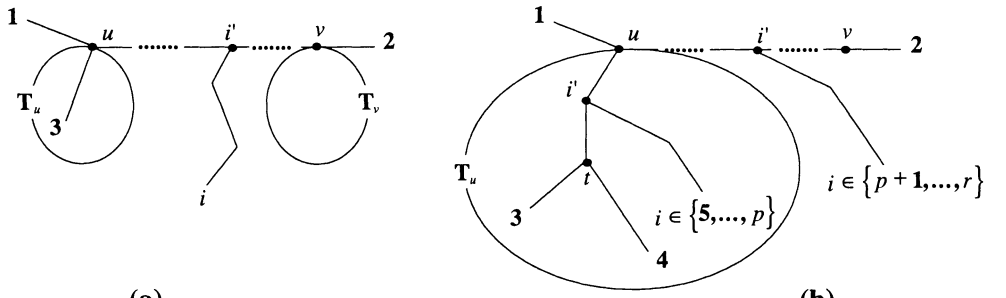
FIGURE 3. **(a)** the interior expression in (7) is equal to the length of the path (i, u); **(b)** when the pair $\{i, j\}$ is examined, d_{tu} is counted thrice and d_{uv} twice; **(c)** when the pair $\{i, k\}$ is examined, d_{tu} is counted once and d_{uv} twice.

Observations made in this example are demonstrated below (4.3) in the general framework. Q is therefore a numerical criterion which, when applied to any tree distance $\mathbf{D}$, designates a pair of neighbors of the tree $\mathbf{T}$ which represent this distance. In practice, we do not make use of the tree distance $\mathbf{D}$, but of its estimation $\mathbf{\Delta}$, and the formula (3) is applied to the estimates δ_{ij} (or λ_{ij}) rather than to the distances d_{ij}. In reality, we thus use an estimator $\hat{Q}$ of the true value of the Q criterion. This estimator Q has an obvious quality which has not been highlighted by Mirkin (1996) yet appears to be very important: it is obtained without making any hypotheses on the δ_{ij} estimates. In fact, it is easy to see (Bulmer 1991) that $(\delta_{xi} + \delta_{yi} - \delta_{xy})$ is the generalized least-squares estimator of the sum $(d_{xi} + d_{yi} - d_{xy})$. In other words, $\hat{Q}$ is the sum of generalized least-squares estimators which each measure the acentrality of the pair to be agglomerated. This does not imply that the estimator $\hat{Q}$ thus obtained is equal, for any given tree structure, to the generalized least-squares estimator of Q. Indeed, for a given tree structure and a given variance-covariance matrix, it is usually possible to find an estimator of Q whose variance is smaller than the variance of $\hat{Q}$. However, we can prove that it is not possible to have an unbiased estimator other than $\hat{Q}$, without already knowing the structure of the tree. Consequently in our context, $\hat{Q}$ seems to be the only possible estimator of Q, and whatever the nature of the variances and covariances of the δ_{ij} estimates, its use appears well founded. We therefore reach a different interpretation from that of Saitou and Nei (1987) and Vach and Degens (1991) who rely on ordinary least-squares, thus losing part of the meaning when the hypothesis of variance identity and covariance nullity is dropped. Thus the use of Q is justified in approaches which drop this hypothesis (Gascuel 1996). The use of Q is also justified within the scope of the hypothesis made in this paper, because after a certain number of steps the estimators λ_{ij} no longer have the same variance due to the mean process (5). Moreover, we can expect a certain robustness of the method, at least concerning its capacity to find the structure of the true tree.

4.3. Proving the correctness of the Q criterion (Property 5). Let $\mathbf{D}$ be a tree distance represented by the tree $\mathbf{T}$ whose valuations are strictly positive, and whose internal nodes are at least of degree 3. Let us consider the Q' criterion $(= Q/2)$ defined above (7), allowing the interpretation proposed in terms of path length of $\mathbf{T}$. Let us suppose that Q' designates the pair $\{1, 2\}$ and that this does not consist of true neighbors in $\mathbf{T}$. The path $(1, 2)$ thus comprises at least two nodes different from nodes 1 and 2. Among these, let us consider nodes u and v, one edge away from 1 and from 2 respectively (Figure 4). Each of these nodes is the root of the subtree denoted $\mathbf{T}_u$, respectively $\mathbf{T}_v$, and two cases can occur: either $\mathbf{T}_u$ or $\mathbf{T}_v$ contains only one leaf (Figure 4a); or these two subtrees each comprise several leaves (Figure 4b). We will demonstrate that in both cases a pair of neighbors exists, whose value for the criterion Q' is strictly greater than the value of the pair $\{1, 2\}$.

Case 3a: Let us suppose that $\mathbf{T}_u$ contains only one leaf (the argument is symmetrical for $\mathbf{T}_v$), and let us consider the notations introduced in Figure 4a. We will demonstrate that the pair of neighbors $\{1, 3\}$ has a better score than $\{1, 2\}$. We have

$$Q'\Big(\{1, 2\}\Big) = d_{12} + d_{3u} + \sum_{i=4}^{r} d_{ii'}.$$



(a) **(b)**

FIGURE 4. **(a)** T_u comprises only the leaf denoted 3; $i \in \{4, \ldots, r\}$ is thus any leaf and i' designates the intersection of the paths $(1,2)$, $(1,i)$ and $(2,i)$. **(b)** T_u comprises a pair of neighbors, denoted $\{3,4\}$, as well as the leaves of index i varying from 5 to p; i' is thus the intersection of paths $(1,3)$, $(1,i)$ and $(3,i)$; the remaining leaf indices vary from $p+1$ to r; i' thus designates the path intersection $(1,2)$, $(1,i)$ and $(2,i)$.

Likewise, we also have

$$\begin{aligned} Q'\Big(\{1,3\}\Big) &= d_{13} + d_{2u} + \sum_{i=4}^{r} (d_{ui'} + d_{ii'}) \\ &= d_{12} + d_{3u} + \sum_{i=4}^{r} (d_{ui'} + d_{ii'}) \\ &= Q'\Big(\{1,2\}\Big) + \sum_{i=4}^{r} d_{ui'}. \end{aligned}$$

The $\mathbf{T}_v$ tree comprises at least one leaf, therefore the last sum is greater than or equal to d_{uv} which, by hypothesis, is strictly positive, and the result is demonstrated.

Case 3b: If $\mathbf{T}_u$ and $\mathbf{T}_v$ each comprise at least two leaves, at least one of these subtrees contains at the most $(r-2)/2$ leaves. Let us suppose that it is $\mathbf{T}_u$ (the argument is symmetrical for $\mathbf{T}_v$). $\mathbf{T}_u$ necessarily contains a true pair of neighbors, denoted $\{3,4\}$. We will demonstrate that this pair is better than $\{1,2\}$. Let us consider the notations introduced in Figure 4b. We have

$$Q'\Big(\{1,2\}\Big) = d_{12} + d_{3u} + d_{4u} + \sum_{i=5}^{p} (d_{ui'} + d_{ii'}) + \sum_{i=p+1}^{r} d_{ii'}.$$

Likewise, we also have

$$\begin{aligned} Q'\Big(\{3,4\}\Big) &= d_{34} + d_{1t} + d_{2t} + \sum_{i=5}^{p} (d_{tu} - d_{ui'} + d_{ii'}) + \sum_{i=p+1}^{r} (d_{tu} + d_{ui'} + d_{ii'}) \\ &= d_{12} + d_{3u} + d_{4u} + \sum_{i=5}^{r} d_{ii'} + \sum_{i=p+1}^{r} d_{ui'} + (r-4)d_{tu} - \sum_{i=5}^{p} d_{ui'} \\ &= Q'\Big(\{1,2\}\Big) + \sum_{i=p+1}^{r} d_{ui'} + A, \quad \text{where} \quad A = (r-4)d_{tu} - 2\sum_{i=5}^{p} d_{ui'}. \end{aligned}$$

As before, we have $\sum_{i=p+1}^{r} d_{ui'} \geq d_{uv} > 0$.

Therefore, all that needs to be shown is: $A \geq 0$. By hypothesis, we know that $\mathbf{T}_u$ contains at most $(r-2)/2$ leaves, including leaves 3 and 4. Moreover, we know that for every $i \in \{5, \ldots, p\}$ we have $d_{ui'} \leq d_{tu}$. Consequently

$$A \geq (r-4)d_{tu} - 2\left(\frac{(r-2)}{2} - 2\right)d_{tu} = 2d_{tu} \geq 0,$$

and the result is demonstrated. ∎

We would like to point out that the case $A = 0$ may occur when the node u is of a degree greater than 3, and when the node t is equal to node u. However, this does not invalidate the proof, which holds when the degree of nodes internal to $\mathbf{T}$ is at least 3. This latter hypothesis is used, effectively, since we assume that the trees $\mathbf{T}_u$ and $\mathbf{T}_v$ each comprise at least one leaf. The case where the internal nodes may be of degree 2 requires special treatment (as well as the redefinition of the notion of neighbor).

5. Least-squares estimation of edge lengths

Sattath and Tversky (1977) were first to find (implicitely) the least-squares estimation of edge lengths of a tree with fixed structure. Brölsch (1983), Vach (1989) and Vach and Degens (1991) have demonstrated a certain number of general properties of this estimation, and they explicitly propose several exact formulae. One of the formulae (5.3) provided by Vach and Degens (1991) is equivalent to formula (4), but more complex due to the use of a reduction formula which differs from (5). It is given below (Equation 10), as well as the proof of its equivalence to formula (4). Likewise, Rzhetsky and Nei (1993) independently found an exact formula which differs from formula (4), yet is identical with another, (4.6), among the formulae of Vach and Degens (1991). This latter formula cannot be integrated into the agglomerative procedure, and it is used in a specific algorithm, in $O(n^3)$, to estimate the edge lengths once the structure has been fully determined.

These studies seem to be relatively unknown and infrequently used, which explains why we try to render this section sufficiently explicit and autonomous. First, in Subsection 5.1, we will describe the fundamental result of Vach (1989) for which we provide a new (to the best of our knowledge) and simple proof. Then, in Subsection 5.2, using this result, we will demonstrate (Property 2) the correctness of formula (4).

5.1. Conservation properties. Let $\boldsymbol{\Delta} = (\delta_{ij})$ be a dissimilarity, $\dot{\mathbf{S}}$ a tree structure, and $\mathbf{S}$ the corresponding adjusted tree (or unsigned tree dissimilarity). Using the notation defined above (Section 2), we have (Vach 1989)

PROPERTY 8. *For every bipartition* $\{X, \overline{X}\}$ *of* $\dot{\mathbf{S}}$, *we have* $s_{X\overline{X}} = \delta_{X\overline{X}}$ *(and* $\overline{s_{X\overline{X}}} = \overline{\delta_{X\overline{X}}}$*).*

In other words, the mean dissimilarity between the elements of X and those of $\overline{X}$ is identical in $\boldsymbol{\Delta}$ and in the adjusted tree $\mathbf{S}$, this being valid for every bipartition of $\dot{\mathbf{S}}$. This property is established very simply from the matrix solution (2). Combining this with equality (1), we obtain directly

$$\mathbf{A}^t\mathbf{S}^t = \mathbf{A}^t\boldsymbol{\Delta}^t. \tag{8}$$

The coefficients of $\mathbf{A}^t$ are expressed as $\mathbf{M}_{k(ij)}$ and have value 1, if and only if the k^{th} edge of $\dot{\mathbf{S}}$ separates i and j. The k^{th} line of the equation (8) thus establishes equality between the sum of the dissimilarities s_{ij} on the one hand, and δ_{ij} on the other hand, provided that i and j are separated by the k^{th} edge. In other words, Property 8 is established for the bipartition associated with the k^{th} edge; and since each bipartition (or edge) of $\dot{\mathbf{S}}$ is represented by a line of $\mathbf{A}^t$, Property 8 is established. ∎

Note that the very same proof applies to any form of additive clustering (Mirkin 1996), where it yields an analogeus conservation property. Moreover, there is another conservation property associated with degree 3 (or ternary) nodes, which has not been referred to by Vach (1989), but which is useful to derive edge length estimates. A ternary node u is the extremity of three edges (u, x), (u, y) and (u, z), and it is associated with the three rooted subtrees X, Y and Z of which x, y and z are the respective roots. We may thus express the following property relative to these subtrees:

PROPERTY 9. *For all ternary nodes of* $\dot{\mathbf{S}}$ *and for every pair* X, Y *of subtrees associated with this node, we have* $s_{XY} = \delta_{XY}$ *(and* $\overline{s_{XY}} = \overline{\delta_{XY}}$*).*

In other words, the mean dissimilarity between the subtrees associated with a ternary node is also preserved. This property is established simply from Property 8. Let X, Y and Z be the three subtrees associated with u. According to 8 we have

$$s_{X\overline{X}} = \delta_{X\overline{X}},\ s_{Y\overline{Y}} = \delta_{Y\overline{Y}} \text{ and } s_{Z\overline{Z}} = \delta_{Z\overline{Z}}.$$

This result and the definition of these quantities enable us to write the three equations

$$\begin{aligned} s_{XY} + s_{XZ} &= \delta_{XY} + \delta_{XZ}, \\ s_{XY} + s_{YZ} &= \delta_{XY} + \delta_{YZ}, \\ s_{XZ} + s_{YZ} &= \delta_{XZ} + \delta_{YZ}, \end{aligned}$$

whose unique solution corresponds to Property 9. ∎

We would like to point out that it is essential to Property 9 that the node be ternary. When u is of degree $g\ (> 3)$, a system of g equations is obtained (one per edge) with $g(g-1)/2$ "unknowns" (one per pair of subtrees), and this system may be satisfied without the mean dissimilarity between subtrees being conserved. It is also worth noting that this result is a generalization of a well known result in the case of ultrametric distances obtained by least-squares adjustment: the mean distance between two neighboring clusters is identical in the observed dissimilarity and in the ultrametric obtained, where it is represented by the level of formation. Finally, let us note that, as with ultrametrics, the dissimilarity between two neighbor leaves x and y linked by a ternary node remains unchanged, *i.e.*, $s_{xy} = \delta_{xy}$.

5.2. Formula (4) is correct (Property 2). The results above directly yield correct least-squares formulae. Let u be a ternary node associated with the three subtrees X, Y and Z whose roots are x, y and z respectively. Using Property 9 and the definitions given above (Section 2), we can write the three equations

$$\begin{aligned} \delta_{XY} &= s_{XY} = n_Y f_X + n_X n_Y (s_{xu} + s_{yu}) + n_X f_Y, \\ \delta_{XZ} &= s_{XZ} = n_Z f_X + n_X n_Z (s_{xu} + s_{zu}) + n_X f_Z, \\ \delta_{YZ} &= s_{YZ} = n_Z f_Y + n_Y n_Z (s_{yu} + s_{zu}) + n_Y f_Z. \end{aligned}$$

And solving these equations, we find

(9) $s_{xu} = \frac{1}{2}\overline{\delta_{XY}} + \frac{1}{2}\overline{\delta_{XZ}} - \frac{1}{2}\overline{\delta_{YZ}} - \overline{f_X}$, s_{yu} and s_{zu} obtained by symmetry

We will now consider an agglomerative procedure as described in Section 3, having as its objective the estimation of the edge lengths of $\dot{\mathbf{S}}$. At each step, the algorithm selects two neighboring edges of $\dot{\mathbf{S}}$, estimates the length of these edges using formula (4), then reduces the matrix using formula (5). At the p^{th} step, it makes use of $r = n - p + 1$ subtrees situated at the "periphery" of $\dot{\mathbf{S}}$ whose edge lengths have already been computed, and all that remains to be estimated is the length of the edges situated at the "center" of $\dot{\mathbf{S}}$. Suppose that we are at step p, and that X and Y are the two trees to be agglomerated. Z represents the "rest" of the tree (whose structure is still unknown if considered from the point of view of UNJ). Taken as a set, Z regroups the objects of the $r - 2$ subtrees different from X and from Y and which have been already resolved, so that we have

$$Z = \bigcup_{I \neq X,Y} I,$$

where I is any one of the subtrees (subsets) already resolved. We may now rewrite the equation (9) as

$$(10) \qquad s_{xu} = \frac{1}{2}\overline{\delta_{XY}} + \frac{1}{2(n - n_U)} \sum_{I \neq X,Y} n_I(\overline{\delta_{XI}} - \overline{\delta_{YI}}) - \overline{f_X},$$

$$\text{where } n_U = n_X + n_Y = n - \sum_{I \neq X,Y} n_I.$$

This formula (10) is correct, and corresponds to formula (5.3) of Vach and Degens (1991). We now need only to show that, at each step of the algorithm, it coincides with formula (4), when the latter is combined with the reduction (5). At the first step, no agglomeration has been achieved, thus $\overline{f_I} = 0$ and $\lambda_{ij} = \overline{\delta_{IJ}} - \overline{f_I} - \overline{f_J}$ for every "subtree" I, J whose respective roots are i and j. It is easy to check that the two formulae coincide. Let us consider the p^{th} step, just before reduction (5), and suppose that

(a) $\lambda_{ij} = \overline{\delta_{IJ}} - \overline{f_I} - \overline{f_J}$ for every resolved subtree I, J whose respective roots are i and j;

(b) formula (4) and equation (10) have coincided during the previous computations.

We will demonstrate that applying the reduction (5) maintains (a), and that (a) being maintained, formulae (4) and (10) coincide during the next step. This means that the hypotheses (a) and (b) are maintained at step $p + 1$ (just before the reduction). By induction, the desired result will follows for all the steps of the algorithm, including the last (the estimation of the last three edge lengths) which is not different from the preceding steps. Let us begin with the first point.

(a) is maintained: As above, the two agglomerated subtrees are denoted X and Y, and they form a new subtree denoted U. We must check that (a) is maintained for the new coefficients λ_{ui} which are obtained by application of reduction (5). We have

$$\begin{aligned} \lambda_{ui} &= w_x\lambda_{xi} + w_y\lambda_{yi} - w_x s_{xu} - w_y s_{yu} \\ &= w_x(\overline{\delta_{XI}} - \overline{f_X} - \overline{f_I}) + w_y(\overline{\delta_{YI}} - \overline{f_Y} - \overline{f_I}) - w_x s_{xu} - w_y s_{yu} \\ &= \overline{\delta_{UI}} - \overline{f_U} - \overline{f_I}. \end{aligned}$$

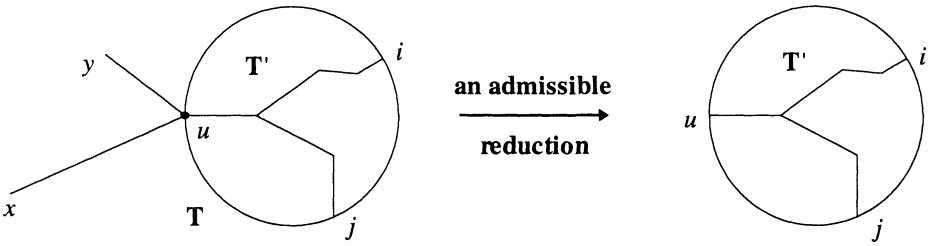


FIGURE 5. An admissible reduction transforms the tree distance represented by $\mathbf{T}$, into the tree distance represented by $\mathbf{T}'$.

The transition from the first to the second line uses (a), whereas to establish the value of $\overline{f_U}$ we have used (b) which supposes, in particular, the correct computation of s_{xu} and s_{yu}. This completes the proof of the first point.

(b) is maintained: For the sake of simplicity, let us again denote as X, Y and U the two agglomerated subtrees and the tree thus formed, even though these are not the same trees as those mentioned above, since we are now at the subsequent step. Utilizing the fact that (a) is maintained, equation (10) can be rewritten in the form

$$\begin{aligned} s_{xu} &= \frac{1}{2}(\lambda_{xy}+\overline{f_X}+\overline{f_Y})+\frac{1}{2(n-n_U)}\sum_{I\neq X,Y} n_I(\lambda_{xi}+\overline{f_X}+\overline{f_I}-\lambda_{yi}-\overline{f_Y}-\overline{f_I})-\overline{f_X} \\ &= \frac{1}{2}\lambda_{xy}+\frac{1}{2(n-n_U)}\sum_{I\neq X,Y} n_I(\lambda_{xi}-\lambda_{yi}), \end{aligned}$$

which corresponds precisely to formula (4), and the proof is completed. ∎

6. Reducing the dissimilarity matrix

In the previous section, we saw that the use of reduction (5) is fully justified from the point of view of the least-squares estimation of the edge lengths. We will now propose a second justification, based on the model placed on the data. First, in Subection 6.1, we show that reduction (5) belongs to a class of "admissible" reduction formulae, in the sense that they guarantee that the true tree to will be found with additive data, when combined with selection criterion (3) and with a correct estimation formula. Then, in Subsection 6.2, we show that among these admissible formulae, reduction (5) corresponds, in some sense, to the minimum variance reduction.

6.1. An admissible reduction formula class. Let us consider Figure 5: given a tree distance $\mathbf{D}$, represented by tree $\mathbf{T}$, the selection criterion (3) systematically designates a pair of neighbors of this tree, denoted $\{x, y\}$, u being the internal node separating these two leaves; then, the estimation formula consists in computing the lengths of the edges (x, u) and (y, u). Let us assume that this formula is correct in the case of additive data, as is the case for formula (4) and for the formula used by NJ, and numerous other possible formulae. We will say that a reduction is *admissible* if it transforms distance $\mathbf{D}$ into distance $\mathbf{D}'$ which is represented by the subtree $\mathbf{T}'$ in which u is now a leaf. Given the properties of the selection criterion and the estimation formula, it is clear that this prerequisite is sufficient to guarantee the finding of the true tree with additive data.

An admissible reduction class is defined by the following generic formula

$$\lambda_{ui} = \mu\lambda_{xi} + (1-\mu)\lambda_{yi} - \mu\hat{d}_{xu} - (1-\mu)\hat{d}_{yu}, \tag{11}$$

where μ is any real number. Indeed, when this reduction is applied to the tree distance $\mathbf{D}$, distances d_{ij} $(i, j \neq x, y)$ remain unchanged, while the dissimilarities λ_{ui} newly introduced satisfy

$$\begin{aligned} \lambda_{ui} &= \mu d_{xi} + (1-\mu)d_{yi} - \mu\hat{d}_{xu} - (1-\mu)\hat{d}_{yu} \\ &= \mu(d_{xi} - d_{xu}) + (1-\mu)(d_{yi} - d_{yu}) \\ &= d_{ui}, \end{aligned}$$

the transition from the first to the second line relying on the fact that the estimation formula is correct, whereas the transition from the second to the third is based on the fact that x and y are neighbors in $\mathbf{T}$.

The NJ reduction formula as formulated by Studier and Keppler (1988) is obtained from expression (11) with $\mu = 1/2$, and that of UNJ with $\mu = n_x/(n_x + n_y)$. An infinite number of other possible solutions exist. The result found here is analogous to that of Bandelt and Dress (1986) on the "convex" versions of the ADDTREE algorithm of Sattath and Tversky (1977). From our point of view, preference for a given NJ version over another version should be based on a data model. As we will see below, this enables us to choose the minimum variance reduction, in other words, the reduction which provides the more reliable estimates to select the pairs of taxa to be agglomerated during the next steps.

6.2. The minimum variance reduction. First of all, it is to be noted that in expression (11) there is a first part $(\mu\lambda_{xi} + (1-\mu)\lambda_{yi})$ which depends on i while the second part $(-\mu\hat{d}_{xu} - (1-\mu)\hat{d}_{yu})$ is identical for every i. Moreover, we can easily demonstrate that if we add a constant k to the reduction formula, then at the following step the selection criterion (3) is increased by $2k$ for every pair $\{x, y\}$, so that the addition of this constant does not affect the choice of the following agglomerations, and does not therefore influence the structure of the tree under construction. Consequently, only the variance of the first two terms has an influence on the structure of this tree.

This variance will be qualified as structural, and for every index i $(\neq x, y)$ it is expressed

$$\mathrm{V}(\lambda_{ui}) = \mu^2\mathrm{V}(\lambda_{xi}) + (1-\mu)^2\mathrm{V}(\lambda_{yi}) + 2\mu(1-\mu)\mathrm{COV}(\lambda_{xi}, \lambda_{yi}). \tag{12}$$

When we try to minimize the sum of the structural variances induced by the reduction, we end up with a second degree polynomial in μ, whose minimum is achieved for

$$\mu^* = \frac{\sum_{\substack{i=1 \\ \neq x,y}}^{t} \Big(\mathrm{V}(\lambda_{yi}) - \mathrm{COV}(\lambda_{xi}, \lambda_{yi})\Big)}{\sum_{\substack{i=1 \\ \neq x,y}}^{r} \Big(\mathrm{V}(\lambda_{xi}) + \mathrm{V}(\lambda_{yi}) - 2\mathrm{COV}(\lambda_{xi}, \lambda_{yi})\Big)}. \tag{13}$$

This is a very general formula. Let us now take into account the characteristics of the chosen data model. We know that the covariance terms are null, and that the variances of the initial dissimilarities δ_{ij} are equal. To simplify, let us assume they are equal to 1. At the first step, we thus have $\mu^* = 1/2$, which corresponds

to reduction (5), and the structural variance of terms λ_{ui}, newly created, is 1/2. We now consider step p, and we suppose that the minimum variance reduction (13) and reduction (5) have coincided up to now. Each index λ_{ij} has thus a structural variance equal to $1/n_i n_j$, with the result that the minimum variance reduction is obtained for

$$\mu^* = \frac{\sum_{\substack{i=1 \\ \neq x,y}}^{r} \left(\frac{1}{n_y n_i}\right)}{\sum_{\substack{i=1 \\ \neq x,y}}^{r} \left(\frac{1}{n_x n_i} + \frac{1}{n_y n_i}\right)} = \frac{n_x}{n_x + n_y},$$

which corresponds to reduction (5). By induction, we deduce that the minimum variance reduction and reduction (5) coincide throughout the algorithm. Moreover, when applying reduction (5) the structural variance is minimized for any index i. It follows that the variance of the estimates $\hat{Q}$ obtained during the further steps will be as low as possible.

Formulae (12) and (13) are very general. Within the scope of our model, they enable us to demonstrate the identity between the minimum variance reduction and reduction (5). However, the main interest of these formulae is elsewhere, this result being widely anticipated. What is interesting is that these formulae lead naturally to a general version of NJ which is able to take into account any variance-covariance matrix of the δ_{ij} estimates. Indeed, formula (13) enables μ^* to be calculated, thus determining the minimum variance reduction. Formula (12), coupled with an analogous formula relating to covariances, enables the same variance-covariance matrix to be updated at each step, so that the process may be repeated iteratively throughout the agglomerative procedure. This is the method we adopted within the scope of a biological sequence data model, and, as expected, significant improvements were obtained concerning the capacity to find the true tree (Gascuel 1996).

7. Simulation results

In order to evaluate the gain obtained by UNJ, within the scope of our model, we conducted simulations based on a schema inspired by Pruzansky *et al.* (1982) and Vach and Degens (1991). These were conducted with the 6 tree structures shown in Figure 6. The first three comprise 12 leaves, while the other 3 comprise 24. We find two extremes in these structures: chains (Chain 12, 24) and perfectly balanced structures (Eql. 12, 24), as well as intermediary structures (Int. 12, 24). In the chains, each (correct) agglomeration consists in adding a unique object to a group which already may comprise several objects. In this case, UNJ and NJ should differ significantly, since the unweighted reduction (5) tends to diverge from the weighted reduction of NJ which is systematically based on $\mu = 1/2$. In the case of balanced structures, each (correct) agglomeration generally agglomerates two clusters comprising the same number of objects, and NJ and UNJ should be extremely close. In the case of intermediary structures, results are expected to be intermediary.

For each of these structures, we generated 500 valued trees by randomly drawing the length of the edges according to a uniform distribution on $[0, 1]$. The tree distance corresponding to each of these valued trees was normalized in order to obtain unit variance. To this normalized tree distance was added a gaussian noise with a null expectation and a standard deviation σ, with $\sigma = 0.1$, 0.3 and 0.6.

Finally, in order to avoid excessively non-metric data, we added a constant to each dissimilarity, so that $\min(\delta_{ij}) = 0.5$.

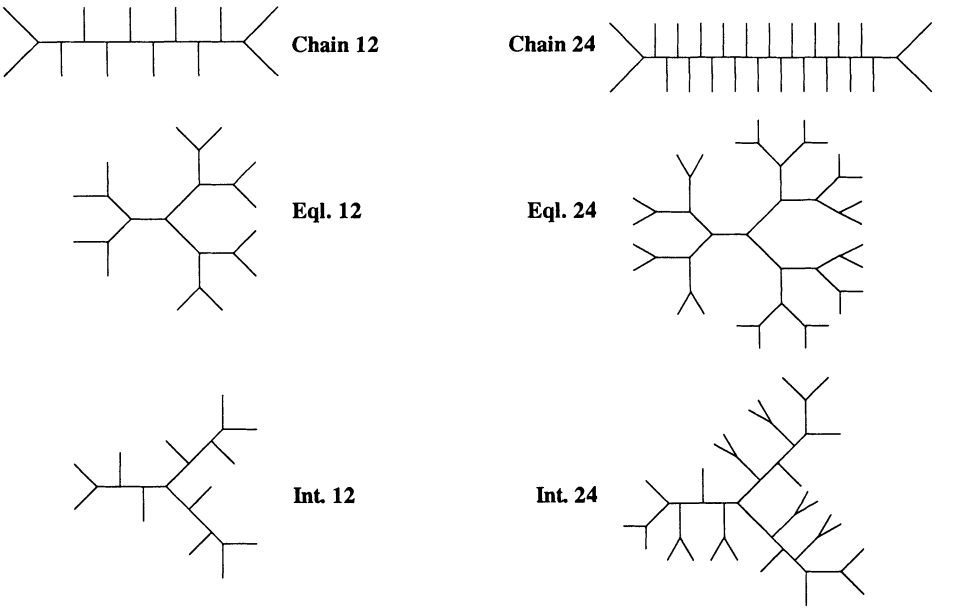


FIGURE 6. The six structures used in the simulations.

For each noisy tree distance obtained, we measured three criteria.

- The Robinson and Foulds (1981) distance between the inferred tree and the true tree. This distance corresponds to the number of bipartitions of the true tree, which are not found by the inferred tree, plus the number of bipartitions of the inferred tree which do not belong to the true tree. Since these quantities are always equal, we show in the table half of this distance, so that for n objects the criterion (RF) lies between 0 (when the trees are identical) and $n - 3$ (when none of the non-trivial bipartitions of the true tree have been found).
- The minimum evolution criterion (ME) which, as explained above, corresponds to the least-squares length estimate of the inferred tree. In the case of NJ (whose estimations are approximate) we used the algorithm described here (Equation 4, Section 3) in order to have the exact value of the criterion. So as to get an idea of the quality of the results obtained, we also applied this criterion to $\dot{\mathbf{T}}$, the structure of the true tree.
- The least-squares criterion (LS), in other words, the squared Euclidean distance between the data matrix $\boldsymbol{\Delta}$ and the inferred tree matrix $\hat{\mathbf{D}}$. Due to the approximate formulae used by NJ, its performances are very poor insofar as this criterion is concerned. Therefore, in the table we show the results obtained by NJ, once the edge lengths have been correctly estimated by (4). Thus we obtain an idea of the quality of the tree structure inferred by NJ. As for the ME criterion, we also applied this criterion to $\dot{\mathbf{T}}$. Moreover, the least-squares criterion partially loses its meaning if we accept negative valuations. In order to relativise the results obtained following this criterion, we also measured, for each tree inferred and for $\dot{\mathbf{T}}$, the number of edges

whose estimated length is negative. The greater their number, the worse the quality of the tree, even if with respect to least-squares, performance is good.

Results are given in Table 1.

- If we consider the capacity (RF) to recover the true tree structure, UNJ is systematically better than NJ. The error reduction is not very high: between about 10% for chains and a moderate noise ($\sigma = 0.1, 0.3$) and 1 to 2% for balanced trees, the reduction for intermediary trees being situated in between. The difference between the number of times where UNJ is better than NJ, and that where it is worse, is sometimes great, reaching 40% in the case of Chain 24 and $\sigma = 0.3$. In all cases, this difference is positive, which proves, if proof is needed, that UNJ should be given preference over NJ when assuming the data model chosen here. Generally, this difference is greater with 24 objects than with 12, which is easily explained because in this case reduction (5) has greater latitude in differing from $\mu = 1/2$, and approaching extreme values 0 and 1. In other words, with a large number of objects, UNJ differs more markedly from NJ, and takes advantage of the specificity of the model. As expected, the method performances decrease considerably when the noise σ increases. Consequently, the absolute gain obtained by UNJ tends to increase, whereas the relative gain tends to diminish.
- If we consider the minimum evolution criterion (ME), the tree inferred by NJ is generally better than the true structure $\dot{\mathbf{T}}$, whereas the tree inferred by UNJ is often better than that of NJ. Given that $\dot{\mathbf{T}}$ is itself likely to be close to the optimum, this proves that in terms of heuristic, UNJ and NJ are very efficient. On the other hand, this capacity to be "better" than $\dot{\mathbf{T}}$ is a handicap since, for example, if a heuristic was systematically better (in this sense) than $\dot{\mathbf{T}}$, it would never find exactly $\dot{\mathbf{T}}$. We thus reach a well known problem in stochastic optimization. Given our results, the solution does not consist in improving the heuristic, but in refining the optimized criterion, so as to make a better selection among the trees close to $\dot{\mathbf{T}}$.
- Similar comments may be made about the least-squares criterion. We note, however, that the domination of NJ on $\dot{\mathbf{T}}$ is weaker than with the ME criterion; this is no doubt due to the fact that once adjusted, $\dot{\mathbf{T}}$ includes a rather large average number of negative edges. We have no explanation for this, and it merits attention in future developments. Otherwise, UNJ largely dominates NJ, while at the same time introducing less negative edges.

8. Discussion

We have presented a second version of NJ, which is unweighted and which we have called UNJ. We have shown that this version is coherent with a data model of the type $(\delta_{ij}) = (d_{ij} + \varepsilon_{ij})$, where (d_{ij}) is a tree distance, and where the ε_{ij} are independent and identically distributed noise variables. This new version derives from the original version of Saitou and Nei (1987) and Studier and Keppler (1988), and also from the results of Vach (1989) and Vach and Degens (1991) concerning the edge length estimation. The simulations show that an appreciable increase has been attained by using UNJ, when the data closely follow the chosen model.

It is not our intention to suggest a systematic preference for UNJ over NJ. Everything depends on the data. With biological sequence data (Swofford *et al.* 1996)

Tree	σ		RF		%RF<,>		%ME<,>		%LS<,>		#NEG	
	0.1	NJ	0.51				36	6	26	15	0.3	0.4
		UNJ	0.47	8	8	5	15	1	13	2	0.2	
Chain 12	0.3	NJ	1.57				73	12	52	32	0.4	0.8
		UNJ	1.42	10	22	10	36	5	34	7	0.2	
	0.6	NJ	3.44				95	4	72	28	0.6	1.6
		UNJ	3.34	3	23	16	47	9	45	10	0.4	
	0.1	NJ	0.33				24	4	20	7	0.5	0.6
		UNJ	0.31	4	4	3	8	1	8	0	0.4	
Int. 12	0.3	NJ	1.18				62	8	50	20	0.5	1.2
		UNJ	1.16	1	10	9	21	3	21	3	0.5	
	0.6	NJ	2.62				91	6	73	24	0.7	1.5
		UNJ	2.55	2	17	11	34	6	32	8	0.6	
	0.1	NJ	0.20				14	3	14	3	0.5	0.6
		UNJ	0.20	1	0	0	1	0	1	0	0.5	
Eql. 12	0.3	NJ	0.80				48	8	45	11	0.5	0.8
		UNJ	0.78	2	3	1	6	0	6	1	0.5	
	0.6	NJ	2.00				74	9	71	12	0.7	1.5
		UNJ	1.97	2	9	5	21	3	19	4	0.6	
	0.1	NJ	2.18				81	11	40	51	0.5	0.8
		UNJ	1.92	12	35	16	53	12	51	12	0.0	
Chain 24	0.3	NJ	7.35				99	1	37	63	1.0	2.5
		UNJ	6.49	12	58	17	73	20	74	19	0.1	
	0.6	NJ	13.3				100	0	47	23	1.2	4.7
		UNJ	12.7	5	50	17	73	23	74	21	0.3	
	0.1	NJ	1.00				50	14	44	17	0.2	0.6
		UNJ	0.92	7	13	6	20	4	18	5	0.1	
Int. 24	0.3	NJ	3.26				87	10	73	24	0.4	3.2
		UNJ	3.04	7	28	13	47	11	44	13	0.2	
	0.6	NJ	7.73				99	1	75	25	0.8	3.1
		UNJ	7.44	4	34	20	69	16	61	19	0.4	
	0.1	NJ	0.67				38	9	37	8	0.2	0.6
		UNJ	0.66	1	2	1	4	1	3	1	0.2	
Eql. 24	0.3	NJ	2.27				78	9	75	12	0.3	1.5
		UNJ	2.24	1	8	6	17	4	15	6	0.3	
	0.6	NJ	5.26				93	5	87	12	0.6	2.6
		UNJ	5.23	1	15	13	39	11	36	12	0.4	

RF is half of the Robinson and Foulds distance between the inferred tree and the true tree; the second item for UNJ indicates the percentage of error reduction obtained when comparing with NJ. **%RF<,>** indicates the percentage of times where UNJ is closer (first item) and farther (second item) than NJ from the true tree. **%ME<,>** refers to the minimum evolution criterion; in the case of NJ, the first item gives the percentage of times where the inferred tree seems better (according to ME) than the true structure $\dot{\mathbf{T}}$, and the second item gives the percentage of times where it seems worse than $\dot{\mathbf{T}}$; in the case of UNJ the items have the same meaning, except that now we compare the inferred tree with that obtained by NJ. **%LS** adopts the same comparison scheme, but for the least-squares criterion. **#NEG** indicates the average number of negative edges; the second item for NJ corresponds to $\dot{\mathbf{T}}$.

TABLE 1. Results obtained for the six structures of Figure 6.

we have noticed, for example, that NJ achieved better performances than UNJ, in terms of ability to recover the true tree structure. This is simply explained by the fact that these data are very far from the model chosen, particularly concerning the hypothesis of independence of the δ_{ij} estimates. However, other data exist which are closer to the model retained, for example, the ADN-ADN hybridization data, based on physical measures and in which the δ_{ij} estimates are basically independent (Felsenstein 1987). For these data and certainly for others, UNJ seems better adapted than NJ.

In fact, our intention is not to "defend" this new version of NJ, but instead to present a framework for the implementation of NJ versions, taking into account the specificity of the data. Within this framework, the model is expressed through the variance-covariance matrix of the δ_{ij} estimates. As we have shown, selection criterion (3) retains its meaning whatever this matrix (Subsection 4.2), and this latter is used at each step to determine the minimum variance reduction (Subsection 5.2). By proceeding in this way throughout the algorithm, we get estimates which are as reliable as possible in choosing the pair to be agglomerated, and we increase the probability of finding the true tree. We have applied this schema here to a very simple classic model. Compared with NJ, the gains observed in the simulations may be qualified as modest, even though they are always positive (Section 7). However, we also applied this same framework to biological sequences (Gascuel 1996). As mentioned above, these data induce variances which vary considerably from one estimate to another, and also give important covariances. Compared with NJ, there is a considerable gain. For certain tree structures, we obtain up to a 50% error reduction, in terms of ability to recover the true tree structure. Generally speaking, we recommend that this approach be used and explored further, notably in the domain of edge length estimation, for which there is no general solution other than the matrix-based technique (Bulmer 1991) which is computationally very expensive.

References

[1] H.J. Bandelt, and A. Dress, *Reconstructing the shape of a tree from observed dissimilarity data*, Advances in Applied Mathematics **7** (1986), 309–343.

[2] H.J. Bandelt, and M. Steel, *Symmetric matrices representable by weighted trees over a cancellative abelian monoid*, SIAM J. on Discrete Math. **8** (1995), 517–525.

[3] J.P. Barthélemy, and A. Guénoche, *Trees and proximity representations*, Wiley, Chichester, 1991.

[4] J. Brölsch, *Minimum-Quadrat-Schätzung von Evolutionsbäumen*, in: I. Dahlberg, M. Schader (eds), Automatisierung in der Klassifikation. Studien zur Klassifikation **13** (1983), 177–187.

[5] M. Bulmer, *Use of the Method of Generalized Least-squares in Reconstructing Phylogenies from Sequence Data*, Mol. Biol. Evol. **8** (1991) 868–883.

[6] P. Buneman, *The recovery of trees from measures of dissimilarity*, in: F.R. Hodson, D.G. Kendall and P. Tautu (eds.), Mathematics in Archeological and Historical Sciences. Edinburgh University Press, Edinburgh, 1971, pp. 387–395.

[7] M.A. Charleston, M.D. Hendy and D. Penny, *Neighbor-Joining uses the optimal weight for net divergence*, Mol. Phyl. Evol. **2** (1993), 6–12.

[8] J.P. Cunningham, *Free trees as representations of psychological distances*, Journal of Mathematical Psychology **17** (1978), 165–188.

[9] G. De Soete, *A least-squares algorithm for fitting additive trees to proximity data*, Psychometrika **48** (1983), 621–626.

[10] J. Felsenstein, *Estimation of Hominoid Phylogeny from a DNA Hybridization Data Set*, J. Mol Evol. **26** (1987), 123–131.

[11] O. Gascuel, *A note on Sattath and Tversky's, Saittou and Nei's and Studier and Keppler's algorithms for inferring phylogenies from evolutionary distances*, Mol. Biol. Evol. **11** (1994), 961–963.
[12] O. Gascuel, and D. Levy, *A reduction algorithm for approximating a (non-metric) dissimilarity by a tree distance*, J. of Classification **13** (1996), 129–155.
[13] O. Gascuel, *BIONJ: an improved version of the NJ algorithm based on a simple model of sequence data*, Les Cahiers du GERAD G-96-59 (1996). To appear in Mol. Biol. Evol. (1997).
[14] K.K. Kidd, and L.A. Sgaramella-Zonta, *Phylogenetic analysis: concepts and methods*, Am. J. Human Genet. **23** (1971), 235–252.
[15] M.K. Kuhner, and J. Felstein, *A simulation comparison of phylogeny algorithms under equal and unequal evolutionary rates*, Mol. Biol. Evol. **11** (1994), 459–468.
[16] S. Kumar, *A stepwise algorithm for Finding Minimum Evolution Trees*, Mol. Biol. Evol. **13** (1996), 584–593.
[17] B. Mirkin, *Mathematical Classification and Clustering*, Kluwer Academic Publishers, London, 1996.
[18] M. Nei, *Relative efficiencies of different tree-making methods for molecular data*, in: M. Miyamoto and J. L. Cracraft (eds.), Phylogenetic Analysis of DNA Sequences, Oxford University Press, Oxford, 1991, pp. 90–128.
[19] S. Pruzansky, A. Tversky, and J.D. Caroll, *Spatial versus tree representation of proximity data*, Psychometrika **47** (1982), 3–19.
[20] D.F. Robinson, and L.R. Foulds, *Comparison of Phylogenetic Trees*, Math. Biosci. **53** (1981), 131–147.
[21] M. Roux, *Techniques of approximation for building two tree structures*, in: C. Hayashi, E. Diday, M. Jambu, and N. Ohsumi (eds.), Recent Developments in Clustering and Data Analysis, Academic Press, New York, 1988, pp. 151–170.
[22] A. Rzhetsky, and N. Nei, *Theoretical Foundation of the Minimum-Evolution Method of Phylogenetic Inference*, Mol. Biol. Evol. **10** (1993), 1073–1095.
[23] N. Saitou, and M. Nei, *The neighbor-joining method: a new method for reconstruction of phylogenetic trees*, Mol. Biol. Evol. **4** (1987), 406–425.
[24] S. Sattath, and A. Tversky, *Additive similarity trees*, Psychometrika **42** (1977), 319–345.
[25] S.R. Searl, *Linear Models*, Wiley, New York, 1971.
[26] P.H.A. Sneath, and R.R. Sokal, *Numerical Taxonomy*, Freeman, San Francisco, 1973.
[27] J.A. Studier, and K.J. Keppler, *A note on the neighbor-joining method of Saitou and Nei*, Mol. Biol. Evol. **5** (1988), 729–731.
[28] D.L. Swofford, G.L. Olsen, P.J. Waddell, and D.M. Hillis, *Phylogenetic inference*, in: D.M. Hillis, C. Moritz and B.K. Mable (eds.), Molecular Sytematics (second edition), Sinauer, Sunderland (MA), 1996, 407–514.
[29] W. Vach, *Least-squares approximation of additive trees*, in: O. Opitz (ed.), Conceptual and Numerical Analysis of Data, Springer, Heidelberg, 1989, 230–238.
[30] W. Vach, and P.O. Degens, *Least-squares Approximation of Additive Trees to Dissimilarities Characterizations and Algorithms*, Computational Statistics Quarterly **3** (1991), 203–218.

Current address: GERAD, École des HEC, 3000, ch. de la Côte-Sainte-Catherine, Montréal (Québec) - H3T 2A7 - CANADA - Until July, 1997
E-mail address: **olivierg@crt.umontreal.ca**

DÉPARTEMENT D'INFORMATIQUE FONDAMENTALE, LIRMM, 161 RUE ADA, 34392 - MONTPELLIER - FRANCE
E-mail address: **gascuel@lirmm.fr**

https://doi.org/10.1090/dimacs/037/10

DIMACS Series in Discrete Mathematics
and Theoretical Computer Science
Volume **37**, 1997

Order Distances in Tree Reconstruction

A. Guénoche

ABSTRACT. The order distance associated with a tree distance is also a tree distance, and the two support trees have compatible topologies. So we are allowed to compute a tree topology from a dissimilarity D using its order distance which only depends on the preorder of the distance values on pairs. In this text we define two methods; the first one is purely ordinal and based on the pair preorder corresponding to D; the second one is a scoring method similar to **ADDTREE**. Computing dissimilarities close to tree distances, we show that these methods permit to determine tree topologies very near to the initial ones.

1. Order distance associated with a tree distance

Let D be a dissimilarity on X and O_x the preorder associated with $x \in X$ that corresponds to the increasing distance from x. Two preorders O_x and O_y disagree about $\{z,t\}$ when $D(x,z) < D(x,t)$ and $D(y,t) < D(y,z)$. There is a semi-agreement when $D(x,z) = D(x,t)$ but $D(y,z) \neq D(y,t)$. The distance $\Delta(O_x, O_y)$ is equal to the number of pairs $\{z,t\}$ for which these preorders disagree, plus half the number of semi-agreements. Δ is the symmetric difference distance between preorder relations [Monjardet 1985]. The order distance D_o associated with D is defined by $D_o(x,y) = \Delta(O_x, O_y)$.

Let us recall that a *pair preorder* is a preorder on the set of pairs of the X elements. A pair preorder can be defined by a distance, pairs being ranked according to the increasing distance values order. Obviously, two distances having the same pair preorder will give the same order distance, and consequently, D_o is the order distance associated with a pair preorder.

To evaluate the distance value between O_x and O_y, for any element u, we count the number of elements that are after u in one preorder and before u in the other one (that are disagreements) plus half the number of pairs that are tied in one preorder and that are not in the other one. Following this algorithm, to compute the order distance on a set of n elements has time complexity $O(n^4)$.

EXAMPLE 1.

Let $O_4 = (4 < 3 < 2 < 1 < 5)$ and $O_2 = (2 < 1 = 4 < 3 = 5)$. We consider elements spanning O_4. Before 4 in O_2 we just have $\{2\}$, then after 3 in O_4 and before 3 in O_2 there are $\{1,2\}$ and that is all; so there are 3 disagreements. There are 2 semi-agreements since we have a single equality between 1 and 4 and between 3 and 5. Consequently $D_o(2,4) = 4$.

1991 *Mathematics Subject Classification.* Primary 05C05; Secondary 05C85, 92B10.

From now on we suppose that D is a tree distance; that is, D satisfies the famous "four points condition": for any $x, y, z, t \in X$,

$$D(x,y) + D(z,t) \leq max\{D(x,z) + D(y,t), D(x,t) + D(y,z)\}$$

In other words among the three sums, $D(x,y) + D(z,t)$, $D(x,z) + D(y,t)$, $D(x,t) + D(y,z)$, the two greatest are equal. This condition is sufficient for a single tree A and a weight function of its edges such that $D(x,y)$ is equal to the length of the path linking x to y, to exist. Two years ago we have presented at the OSDA'95 conference the following theorem:

THEOREM 1 (Bonnot, Guénoche, Perrier 1996). *If D is a tree metric that can be represented by A as a set of edges, its order distance D_o is also a tree metric that can be represented by some $A_o \subset A$.*

Knowing A and the edge's lengths, and consequently D, it is easy to build manually A_o and so to calculate D_o. Each pair $\{x, y\}$ of elements of X gives one weight unity to the edges of A according to the position of the middle $M(x,y)$ of the path between x and y. If $M(x,y)$ is the common end of two edges, each one receives 1/2 point, and if $M(x,y)$ is between the two ends of a single edge, it receives the whole point. The edges of A_o are those that have a positive total weight; those having a weight equal to zero vanish from the support tree of D_o. And so an edge that contains a point at equal distance from two elements of X on both sides of this edge is kept in A_o; an edge that does not contain a middle point is contracted. The distance value $D_o(x,y)$ is the sum of the weights along the path between x and y.

EXAMPLE 2.

Let A be the support tree of the distance D shown in Figure 1. The nodes unlabelled by X are numbered $\{1, 2\}$, and the edge's lengths are given in bold. The tree A_o corresponds to the order distance D_o associated with D. The edge's length in A_o are the row sums in Table 1. The edges $(1, u)$ and $(2, z)$ in A have null length in A_o; vertices 1 and 2 have disappear.

D	x	y	z	t
y	4			
z	4	4		
t	5	5	3	
u	3	3	3	4

D_0	x	y	z	t
y	4			
z	6.5	6.5		
t	8	8	1.5	
u	2	2	4.5	6

	xy	xz	xt	xu	yz	yt	yu	zt	zu	tu	
(1,x)	1/2	1/2	0	1	0	0	0	0	0	0	**2**
(1,y)	1/2	0	0	0	1/2	0	1	0	0	0	**2**
(1,u)	0	0	0	0	0	0	0	0	0	0	**0**
(1,2)	0	1/2	1	0	1/2	1	0	0	1	1/2	**9/2**
(2,z)	0	0	0	0	0	0	0	0	0	0	**0**
(2,t)	0	0	0	0	0	0	0	1	0	1/2	**3/2**

TABLE 1. Contributions of pairs of X to the weight of the edges in A_o

So the tree topology of the order distance associated with a tree distance can be weaken, since some edges can be contracted. It follows from the above that

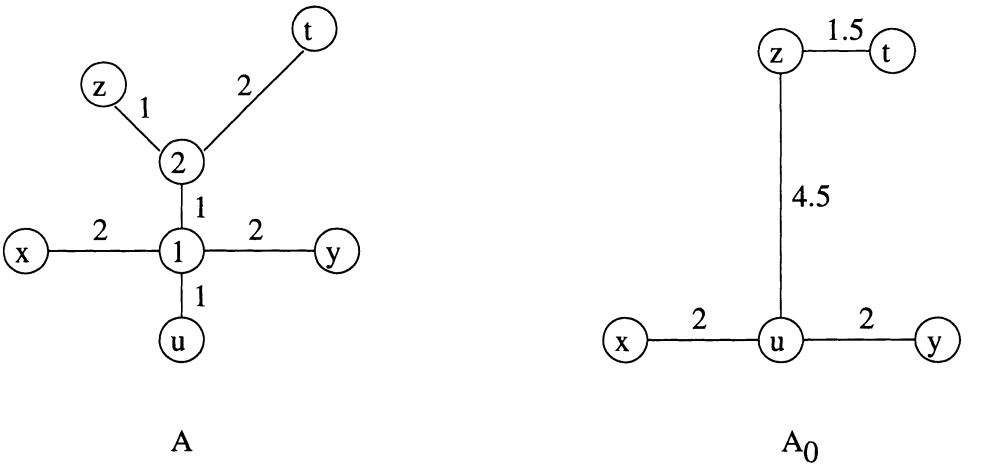


FIGURE 1. Trees A and A_0.

supporting tree of an ultrametric does not change at all, which was differently stated in Guénoche [1994]. For an ordinary tree distance, the shortest of the two connected edges linking two adjacent leaves is lost; some inner edges can also disappear. But for most quadruples their topologies are the same in A and A_o. Those that are weaken have lost their inner edge.

2. Constructing a tree using order distances

In phylogenetic reconstruction [Darlu & Tassy, 1993], the tree model is basic. Since Darwin and Linnaeus, taxonomic units are derived along a tree, the root of which being the common ancestor. These trees are frequently established using phenetic methods based on distance. The values are computed from morphological or molecular data that are difficult to evaluate. One cannot know if an attribute is important or if it has not been counted previously. In DNA sequences, comparing characters on sites after a multiple alignment, one can ask if different nucleotides correspond to a single or multiple mutation; more, two identical bases can hide a reversal mutation. Briefly speaking, we are generally doubtful of distance values. But small differences can lead to fundamental mistakes, as shown in Example 3.

EXAMPLE 3.

We consider 3 tree distances very close together that correspond to the 3 differentiated possible topologies for 4 elements:

	x	y	z
y	9		
z	8	16	
t	15	23	17

$T_1 = \{x, y \mid z, t\}$

	x	y	z
y	12		
z	6	16	
t	14	19	18

$T_2 = \{x, z \mid y, t\}$

	x	y	z
y	11		
z	9	13	
t	12	21	19

$T_3 = \{x, t \mid y, z\}$

They give the same order distance, since for each one O_x, O_y, O_z and O_t are identical. Constructing a topology from D_o, we find $D_o(x,y) + D_o(z,t) = D_o(x,z) + D_o(y,t) = D_o(x,t) + D_o(y,z)$, and the topology $T_0 = \{x, y, z, t\}$ having no differentiated pair.

O_x = (x<z<y<t)
O_y = (y<x<z<t)
O_z = (z<x<y<t)
O_t = (t<x<z<y)

D_o	x	y	z
y	2		
z	1	3	
t	3	5	4

In this text we want to measure the structural difference between a tree computed from order distance, a tree computed from distance values and a true tree. We do that simulating data close to the tree model. We start from a tree distance D and its support tree A. Then we evaluate a distance D' close to D and compute its order distance D'_o. Using the ADDTREE method of Sattath & Tversky [1977], we get two trees, A' directly from D' and A'_o from D'_o. We are only interested in tree topologies, corresponding to the edge's lists, because for evolution problems, the set of bifurcations is primordial. More, knowing the tree structure, edge's lengths can be evaluated minimising a least square criterion [Barthélemy & Guénoche 1991].

2.1. A purely ordinal method. One starts from distance D' obtained noising a tree distance D. We first calculate its order distance D'_o and we build the tree using only these values. As D'_o only depends on the preorder of the values of D', one can use as data the pair preorder of D'.

Example 3 proves that this method does not provide systematically the initial topology; the support tree of D'_o may be not complete, and some nodes have degree greater than 3. In this specific case it can be considered as a benefit, if we admit that none differentiated topology is evident.

We also want to underline that D'_o can be a tree distance, even if D' is not; it is obvious if D' has the same pair preorder than D. In that case A'_o does not depend on the method used to construct it.

2.2. The ordinal scoring method. The ADDTREE method is founded on the following principle: The tree topology is related to quadruple's topologies, because they indicate pairs that must be joined together in first. For that, each quadruple gives score points to differentiated pairs, and the selected pair is the one that get the largest number of points. In our ordinal scoring method, we use the same principle, but score points are given according to the order distance relative to each quadruple $\{x, y, z, t\}$ which is computed from 4 preorders O_x, O_y, O_z, O_t reduced to the 4 elements $\{x, y, z, t\}$. This 4-points order distance indicates the *ordinal topology* of this quadruple; it is the tree topology minimising the sum of order distances between differentiated pair.

In Example 3, D_o is a tree distance since the 3 sums are equal. When D is not a tree distance, D_o can have two sums equal and lower than the third one. In that case we decide that there are two ordinal topologies. Studying systematically all the total orders on distance values between four elements, we have observed that 346/720, that is practically 1/2, have an order distance that is a tree metric [Guénoche 1997].

Doing the same as for ADDTREE, we give score points to pairs that realise an ordinal topology. As we want that any quadruple give the same number of points, and as we may have 2, 4 or 6 pairs to credit, each quadruple gives 12 points (to have integer score values). If the ordinal topology is unique, for instance T_1, we give 6 points to (x, y) and 6 points to (z, t). If they are two, for instance T_1 and T_2, pairs $(x, y), (z, t), (x, z)$ and (y, t) receive 3 points. If the ordinal topology is T_0, the 6 involved pairs get 2 points.

ORDINAL SCORING ALGORITHM

One step consists in:

1. For each quadruple $\{x, y, z, t\}$
   (a) Compute its order distance restricted to these elements,
   (b) Add score points to pairs involved in an ordinal topology
2. Select the pair $\{x, y\}$ having maximum score
3. Remove $\{x, y\}$ and add a new vertex w and two edges (x, w) and (y, w)
4. Evaluate the edge's lengths with the usual formula
5. Update the distance matrix

In fact it is an hybrid method, because distance values are needed to compute ordinal topologies and to determine which is the pair to join first. As for ADDTREE, this algorithm is in $O(n^5)$, because order distances on quadruples can be calculated in constant time ($O(1)$). Nevertheless it is a time consuming procedure.

3. Protocol for simulations

To realise computer simulations we need:

- distances close to random tree metrics,
- parameters to compare topologies of A' and ${A'}_o$ to A.

3.1. Distances close to tree metrics. We want to check methods with random X-tree topologies. When hierarchies are selected at random, expending a labelled tree from its leaves [Lapointe & Legendre, 1991], it is hard to assume that all the possible topologies are tested. We now give a method to generate at random binary unlabelled rooted trees (BURT) and to generate all of them. We first put an order on such trees and count them.

Let $\boldsymbol{A}$ be the set of BURTs and $\boldsymbol{A}_n$ be the set restricted to trees with n leaves. Each tree A_i is the union of two subtrees A_i^g on the left side and A_i^d on the right side. We now define a total order (denoted $\prec$) on $\boldsymbol{A}$:

- Any tree with n leaves is before a tree with m leaves if $n < m$.
- Two trees A_i and A_j having the same number of leaves verify:

$$A_i \prec A_j \Leftrightarrow A_i^g \prec A_j^g \text{ or } A_i^g = A_j^g \text{ and } A_i^d \prec A_j^d$$

In Figure 2 all the BURTs with at most six leaves are ranked according to this order.

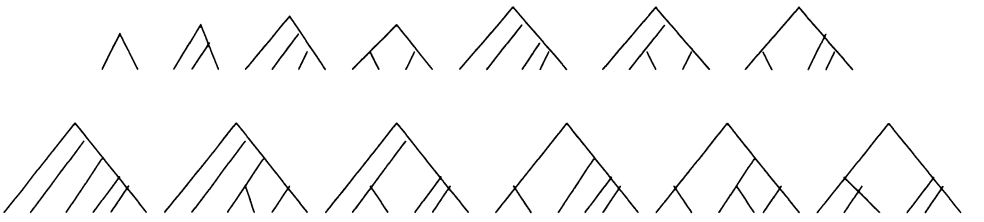


FIGURE 2. The 13 first BURTs.

This order imposes that the left subtree is never greater than the right one. This permits to count BURTs easily and also to select them at random. Let T_n be the number of BURTs with n leaves.

PROPOSITION 1. *T_n is given by the following recurrence formula:*

$$T_1 = T_2 = 1 \text{ then } T_{2k+1} = \sum_{i=1}^{k} T_i T_{2k+1-i} \text{ and } T_{2k} = \sum_{i=1}^{k-1} T_i T_{2k-i} + \frac{1}{2} T_k (T_k + 1)$$

Proof

If n is odd, let $n = 2k + 1$. The left subtree may have $i \leq k$ elements and the right one $2k + 1 - i$. If n is even, $n = 2k$; we count first trees having a left subtree with no more leaves than the right one. Then we count those that are balanced; their number is equal to the number of pairs, maybe with identical elements, in A_k.

Computing the number of BURTs, we get the Table 2:

n	3	4	5	6	7	8	9	10	11	12	13	14
T_n	1	2	3	6	11	23	36	98	207	451	983	2149

Table 2: The numbers of bianary unlabelled rooted trees with n leaves.

Now to build a random BURT with n leaves, we come back to an old method developped in our PhD [Guénoche 1979]. The cardinality of A_n being known, and A being properly ordered, it suffices to select at random an integer $1 \leq R \leq T_n$ and to build the BURT having this rank according to the given order. If R is selected uniformely, any tree has an equal probability. To determine the number of leaves in its subtrees, it is sufficient to compute

$$\Sigma_i = T_1 T_{n-1} + T_2 T_{n-2} + .. + T_i T_{n-i}$$

until to reach R. The index value i is the smallest integer such that:

$$\Sigma_{i-1} < R \leq \Sigma_i$$

Consequently the left subtree possesses i leaves and the right one has $n - i$. Just counting, we have jumped in A_n Σ_{i-1} trees, and the searched tree with rank R is exactly the one with rank $R' = R - \Sigma_{i-1}$ among those having a left subtree with i leaves.

Until now, it is not necessary to distinguish for n by parity, because, if $n = 2k$ we have $\Sigma_k > T_n$ and the balanced trees are placed at the end. But to calculate, from R, the ranks R_g and R_d of the left and right subtrees repectively in A_i and A_{n-i}, we must count differently according to n, since balanced trees occur only when n is even.

Case $n = 2k + 1$: : For each one of the T_i left subtree, there are T_{n-i} trees having this decomposition. It is again sufficient to add classes of T_{n-i} trees until to reach or overpass R'. If we note $\lfloor x \rfloor$ the integer that is not greater than x, we have:

$$R_g = 1 + \lfloor \frac{R'}{T_{n-i}} \rfloor \text{ and } R_d = R' - (R_g - 1) T_{n-i}$$

Case $n = 2k$: : If the left and right subtrees do not have the same number of leaves, then $i < n - i$ and the formulas given above are correct. But if the searched tree is balanced, $R' = R - \Sigma_{k-1}$, and we are in the last class of trees corresponding to the last term in the formula for T_{2k}. *To* respect the given order on BURTs, we must have $R_g \leq R_d \leq T_k$. To evaluate R_g, one must

compute sums: $s_0 = 0, s_1 = T_k, s_2 = s_1 + T_k - 1, .., s_j = s_{j-1} + T_k - j + 1$ until to reach R'. Consequently we have $s_{j-1} < R' \leq s_j$. Then

$$R_g = j \text{ and } R_d = R' - s_{j-1}.$$

Doing so we have substituted the initial problem of finding the BURT with rank R in $\boldsymbol{A}_n$ with two smaller problems of the same type. The solution beeing evident when $R = 1$ or when $n \leq 3$, this algorithm converges.

EXAMPLE 4.

At the first step, we build the BURT with rank $R = 58$ in $\boldsymbol{A}_{10}$. We have $\Sigma_1 = T_1T_9 = 36, \Sigma_2 = T_1T_9 + T_2T_8 = 59$, and so this tree has 2 leaves in its left subtree (and 8 on the right side). We get $R' = 58 - 36 = 22$. As $T_8 = 23$ we obtain $R_g = 1$ and $R_d = 22$.

At the second step, we build on one side the tree with rank 1 in $\boldsymbol{A}_2$ (it is easy since there is only one), and on the other side the tree with rank $R = 22$ in $\boldsymbol{A}_8$. We have $\Sigma_3 = 20$ and $\Sigma_4 = T_1T_7 + T_2T_6 + T_3T_5 + T_4T_4 = 24$, hence this tree has 4 leaves in its both subtrees; we tackle the specific part of the even case. We have $R' = 22 - 20 = 2$. As $T_4 = 2$ we have $R_g = 1$ and $R_d = 2$. Both subtrees are respectively the first one and the second one in $\boldsymbol{A}_8$.

Finally the 58-th BURT with 10 leaves, is drawn in Figure 3, between those having ranks 57 and 59.

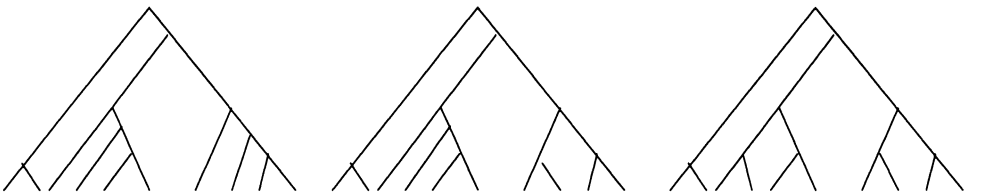


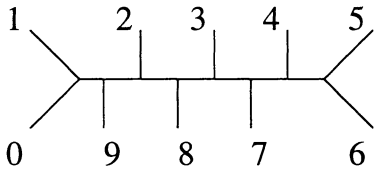
FIGURE 3. BURTs with ranks 57, 58 and 59 in $\boldsymbol{A}_{10}$.

Now we have random binary unlabelled rooted trees, that correspond to random phylogenic topologies. Forgeting the root we obtain an X-tree A which can be coded in a binary table with $|X| = n$ rows and as much columns as the number of edges in A. It is well known that an edge corresponds to a partition of X in two classes X_1 and X_2, composed of the elements of X on its both sides. For each one, in the table, we shall give to the smallest class value 1 and to the largest one value 0; this arbitrary choice has no importance when computing distance.

EXAMPLE 5.

Labelling arbitrarily the X-tree of Figure 4, we get the table as indicated below. Each column corresponds to a bipartition. The first ten are for the edges ending in X and the others are for the seven inner edges.

We now give some random weights to these columns to compute the Hamming distance between rows and to obtain a tree metric with this topology. Starting from this table, and a tree distance D, one can obtain another distance D' close to D applying two different operations:



1: 1 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0
2: 0 1 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0
3: 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
4: 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 1 0
5: 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 1 1
6: 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 1 1
7: 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0
8: 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0
9: 0 0 0 0 0 0 0 0 1 0 0 1 1 1 0 0 0
0: 0 0 0 0 0 0 0 0 0 1 1 1 1 1 0 0 0

FIGURE 4. A tree and the corresponfing bipartition table.

- Directly, by adding or subtracting to each $D(x, y)$ a percentage ε of this value. This ε is bounded by the maximum degree of noise τ, which is a parameter for simulations. For any pair, ε is uniformly selected at random between 1 and τ. So $D'(x, y) = (1 \pm \varepsilon).D(x, y)$. Some values are larger, others smaller, the triangular inequality is not necessarily kept; it is very rough.
- Indirectly, by modifying values in the table corresponding to a topology. We add or subtract a quantity ε, bounded by τ, which is also the percentage of values to modify. Then we compute again the Hamming distance, with the same weights, but fuzzyfying the table we do not get anymore a tree metric.

The first kind of noise corresponds to the choice of an inadequate metric function on X; sometimes values are over estimated and sometimes under estimated. The second kind of noise corresponds to a possible erroneous information to evaluate distance; it is the general case for evolution problems.

To obtain dissimilarities close to tree metrics, we modify:

- distance values: For each random tree, we select at random edge's weights, compute a tree metric and we noise it using $n(n-1)/2$ random ε as indicated above.
- table values: For each random tree, we randomly select cells in the table that will be altered, and we choose at random column's weights to compute Hamming distance.

3.2. Parameters to compare tree topologies. To evaluate the gap between the topologies of A and A', or A'_o, we define two parameters, the percentage of quadruples that are correctly represented, and the percentage of specific bipartitions.

- According to the A tree, for any quadruple $\{x, y, z, t\}$, there are four possible topologies. For three of them there is an inner edge to separate two differentiated pairs, for instance $\{x, y \mid z, t\}$. For the last one, the six paths share a

common point; we shall say that this topology is *weaken*. According to the other tree, A' or A'_0, for this same quadruple we have three possibilities:

- The topology is the same as in A, weaken or not. So quadruple $\{x, y, z, t\}$ is correctly represented; we count one unity.
- Only one of the two topologies is weaken, whatever it is A or A'. The inner edge has vanished and we count $1/2$ unity.
- Topologies are both differentiated and different; we do not count anything.

Parameter T_q denotes the number of unity divided by the number of quadruples. The larger it is, the better is the topology.

- An edge separates X in two classes. If it has one end in X, one of these classes is a singleton. We shall name *external* this kind of edge, in contrast with the inner edges having both ends with degree at least 3. Comparing A and A', we find common edges corresponding to the same bipartitions, and edges that do not have an equivalent in the other tree; they are specific of one topology. In most cases, A' is a complete X-tree and there is $|X|$ external edges shared with A. But for A'_0, some of them have disappeared, because they do not contain the middle of any path. Not to disadvantage methods based on order distances, we only count the number of specific bipartitions corresponding to inner edges.

  This quantity is divided by the number of inner edges in both trees. Let ΔP be this second parameter. It is a symmetrical difference distance between trees, after contraction of the external edges. If we had taken under consideration all the bipartitions, corresponding to inner and external edges, it would be the distance introduced by Robinson & Foulds [1981].

EXAMPLE 6.

In the two trees of Figure 5, there is only one quadruple $\{x, z, t, u\}$ having the same topology. Quadruples $\{x, y, t, u\}$ and $\{y, z, t, u\}$ have the weaken topology in the tree on the right side. The two remaining quadruples $\{x, y, z, t\}$ and $\{x, y, z, u\}$ have different and differentiated topologies in both trees; consequently $T_q = 2/5$

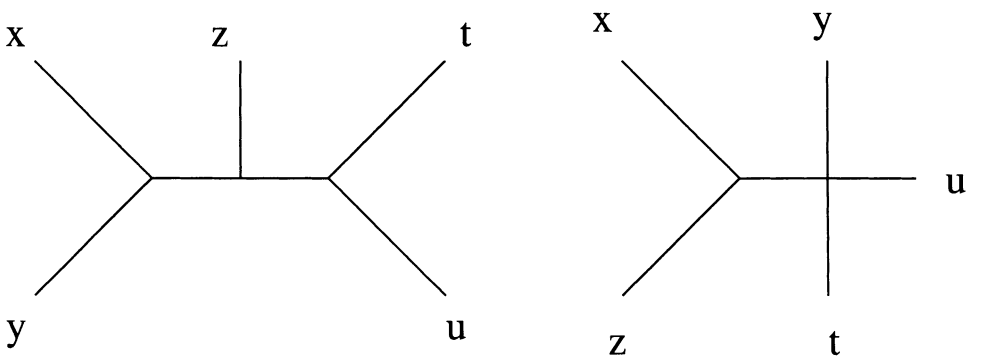


FIGURE 5

The tree on the left side has two inner bipartitions $\{x, y \mid z, t, u\}$ and $\{x, y, z \mid t, u\}$. The tree on the right side has only one $\{x, z \mid y, t, u\}$; All the

three are specific and so $\Delta P = 1$. If we had applied the Robinson - Foulds formula, one would have found 5 common bipartitions and a tree distance equal to 3/13.

4. Results

For each value of τ, we have tested the 98 BURTs with 10 leaves, corresponding to the 11 different X-trees with $|X| = 10$ drawn in annex. In front of each tree, between brackets, we have indicated the number of corresponding rooted trees.

4.1. Ordinal method. MODIFICATIONS ON DISTANCE VALUES

	T_q		ΔP	
	A'	A'_o	A'	A'_o
$\tau = 0$	1	.98	0	.07
$\tau = .05$	.98	.94	.04	.15
$\tau = .1$	.96	.92	.07	.20
$\tau = .15$	.94	.88	.12	.25

MODIFICATIONS ON TABLE VALUES

	T_q		ΔP	
	A'	A'_o	A'	A'_o
$\tau = 0$	1	.98	0	.07
$\tau = .05$	.99	.96	.02	.09
$\tau = .1$	.98	.95	.05	.13
$\tau = .15$	.96	.92	.09	.18

4.2. Ordinal scoring method. MODIFICATIONS ON DISTANCE VALUES

	T_q		ΔP	
	A'	A'_o	A'	A'_o
$\tau = 0$	1	.99	0	.07
$\tau = .05$	.99	.97	.03	.10
$\tau = .1$	.96	.94	.09	.17
$\tau = .15$	.93	.92	.13	.19

MODIFICATIONS ON TABLE VALUES

	T_q		ΔP	
	A'	A'_o	A'	A'_o
$\tau = 0$	1	.99	0	.07
$\tau = .05$	.99	.97	.02	.09
$\tau = .1$	.98	.95	.04	.14
$\tau = .15$	.95	.92	.09	.19

Following this protocol, we have compared the tree topologies obtained from distance and from order distances. The first ones are always slightly better ; it means, that trees computed from distance values are structurally closer to the true tree than those computed using order distance values.

Nevertheless, if distance values are not available, and if you just know the pair preorder, you can get a tree topology that is very close to the one of the true tree. It is obvious reading parameter values when $\tau = 0$.

We bring to the end, underlying that the ordinal scoring method is better than the simple ordinal method. It is not surprising after the observation made: it is more efficient to use distance rather than pair preorders, even if you are only interested in tree topology.

We have drawn the eleven X-trees with ten leaves that have a maximum number of edges. They correspond to all the different unlabelled topologies of unrooted trees. Between parentheses is the number of BURTs having this unrooted topology; it is the number of different ways to root them.

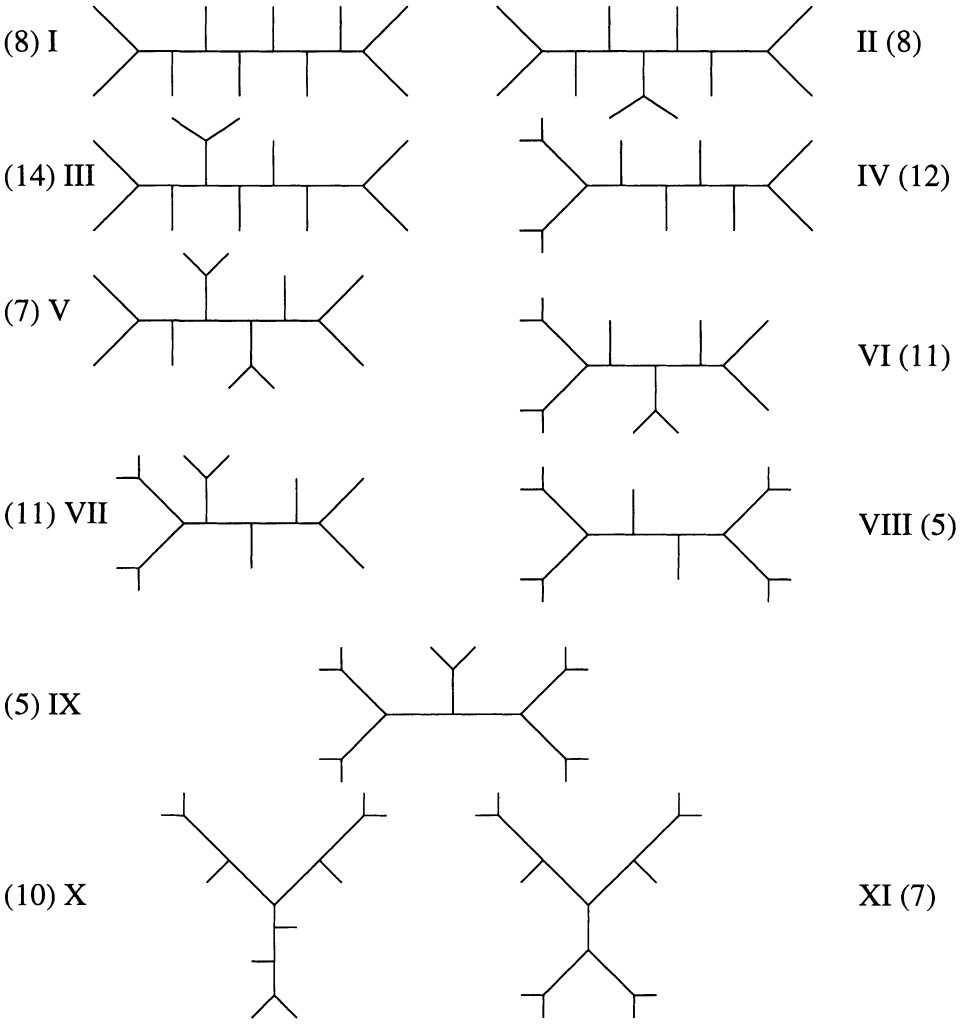


FIGURE 6

References

[1] J.P. Barthélemy, A. Guénoche (1991) *Trees and Proximity Representation*, J. Wiley.
[2] F. Bonnot, A. Guénoche, X. Perrier (1996) Properties of an order distance associated to a tree distance, *Proceedings of "Ordinal and Symbolic Data Analysis"*, Springer, pp. 252-261.
[3] P. Darlu, P. Tassy (1993) *Reconstruction phylogénétique*, Masson.
[4] A. Guénoche (1979) *Algorithmes d'énumération d'ensembles combinatoires.* Doctorat de spécialité, Université de Provence.
[5] A. Guénoche (1994) Order distance associated with a hierarchy, Conference "Combinatorics in Behavioral Sciences", Irvine, August 1994, *Journal of Classification*, to appear.
[6] A. Guénoche (1997) Ordinal properties of tree distances, submitted for publication.
[7] J.F. Lapointe, P. Legendre (1991) The generation of random ultrametric matrices representing dendrograms, *Journal of Classification*, 8, pp. 177-200.
[8] B. Monjardet, (1985) Concordance et consensus d'ordres totaux : Les coefficients K et W, *Revue de Statistique Appliquée*, 33, 2, pp. 55-87.
[9] D.F. Robinson, L.R. Foulds (1981) Comparision of phylogenetic trees, *Math. Biosc.*, 53, pp. 33-40.
[10] S. Sattath, A. Tversky (1977) Additive similarity trees, *Psychometrika*, 42, 319-345.

LIM - CNRS, 163 Av. de Luminy, 13288 Marseille Cedex 9
E-mail address: **guenoche@lim.univ-mrs.fr**

https://doi.org/10.1090/dimacs/037/11

DIMACS Series in Discrete Mathematics
and Theoretical Computer Science
Volume **37**, 1997

Circular orders of tree metrics, and their uses for the reconstruction and fitting of phylogenetic trees

Vladimir Makarenkov and Bruno Leclerc

ABSTRACT. The circular orders associated with the planar drawings of an X-tree (or phylogenetic tree) have been studied by several authors. They allow an encoding of an X-tree by $2n$-3 numbers (where n is the number of elements of X), the lengths of some paths between leaves of the tree. It is shown here that circular orders are the same as those obtained from the table of a tree metric by a construction due to Yushmanov [28]. It is also observed that this construction applies to any dissimilarity, tree metric or not. Several fast algorithms (of complexity $O(n^2)$) are derived from these results: for the determination of a Yushmanov order; for the reconstruction of the valued X-tree represention of a tree metric; for the recognition of a tree metric; and for the fitting of a tree metric to a given dissimilarity; the fitting method is based on successive local least squares approximations. Tested on various experimental and real data, it gives satisfactory results.

РЕЗЮМЕ. Многие авторы подробно исследовали циркулярные порядки соответствующие графическому представлению X-дерева (или филогенетического дерева), в частности в случае его кодирования посредством $2n$-3 значений расстояний между его листьями (где n представляет собой число элементов множества X). В настоящей статье мы покажем, что циркулярные порядки являются эквивалентными порядкам, введённым Юшмановым в [28], которые могут быть получены непосредственно из матрицы расстояний. Более того, эти порядки могут быть также определены для произвольной матрицы коэффициентов различия. Полученные результаты позволяют вывести несколько интересных алгоритмов (сложности $O(n^2)$ операций), среди которых алгоритмы: для определения порядка Юшманова, для восстановления дерева по матрице расстояний, для распознавания древесной структуры и для аппроксимации данной матрицы различий древесной метрикой. Последний алгоритм , состоящий из последовательных аппроксимаций, согласно критерию наименьших квадратов, даёт разумные результаты в примерах, на которых он был тестирован.

RÉSUMÉ. Plusieurs auteurs ont étudié les ordres circulaires associés aux représentations planaires des X-arbres (ou arbres phylogénétiques), en particulier pour le codage d'un X-arbre valué par $2n$-3 longueurs de chemins d'une feuille à une autre (n étant le nombre d'éléments de X). On montre dans cet article que ces ordres circulaires sont les mêmes que ceux obtenus à partir de la table d'une distance d'arbre par une construction due à Yushmanov [28]. De plus, cette construction s'applique à toute dissimilarité,

1991 *Mathematics Subject Classification.* Primary 05C05; Secondary 05C85, 92B10.
This research was partly supported by Esprit LTR Project n° 20244-ALCOM-IT. The authors are indebted to the Département Informatique and the Centre Documentaire of the École Nationale Supérieure des Télécommunications de Paris for support in bibliographic research and programming work. They also wish to thank Alain Guénoche for helpful advice and comments.

distance d'arbre ou non. Ces résultats permettent d'obtenir plusieurs algorithmes rapides (de complexité $O(n^2)$) : pour la détermination d'un ordre de Yushmanov ; pour la reconstruction du X-arbre valué représentant une distance d'arbre ; pour la reconnaissance d'une distance d'arbre ; et pour l'ajustement d'une distance d'arbre à une dissimilarité d donnée. Ce dernier comporte des approximations locales successives par les moindres carrés. Il conduit à une procédure d'ajustement qui donne des résultats intéressants sur les exemples sur lesquels elle a été testée.

1. Introduction

Let X be a finite set with n elements. We consider a tree T with leaves labelled according to X. When T is endowed with a non-negative edge length function, a *tree metric* d on X is defined as follows: for all $x, y \in X$, $d(x,y)$ is the length of the unique path of T between the leaves x and y.

The classical representation of a metric on X by the matrix of its values for all pairs of elements of X is fairly redundant in the case of a tree metric. Two ways of summarizing a tree metric by $2n$-3 entries, the minimum possible number, are presented in Leclerc [17]. Thay are both based on minimum spanning trees; previously, Barthélemy and Guénoche [2], generalizing a result of Chaiken, Dewdney and Slater [7], have shown the entire matrix of a tree metric d to be defined by its restriction to another set R of $2n$-3 entries associated with a so-called *diagonal plane order* on X. Diagonal plane orders are defined in geometrical terms, in relation with the planar drawings of T. On the contrary, the knowledge of such a graphical representation is not needed for the determination of the linear orders defined by Yushmanov [28], together with another way of summarizing a tree metric by $2n$-3 entries; Yushmanov orders are directly obtained from the matrix of d.

It is shown here in Section 3 that circular and Yushmanov orders are in fact exactly the same. This paper is devoted to the study of these orders, and their use in an approach of the problems of recognition, reconstruction and fitting of tree metrics.

The paper is organized as follows. Basic definitions are recalled in Section 2.1; Section 2.2 includes a synthetic presentation of the diagonal and circular orders associated with a valued X-tree, with some of their properties. Although the results of this section are not new, some of them are given in new statements, more complete than the previous ones, and with simplified proofs. Yushmanov's results, with the presentation of his orders, are recalled in Section 3.1. It is emphasized that the definition of Yushmanov orders extends to all dissimilarities. Two algorithms are given, the first one for the determination of a Yushmanov order and the second for the reconstruction of the valued X-tree associated with an initial tree dissimilarity. After an illustration of these algorithms in Section 3.2, it is shown in Section 3.3 that Yushmanov and circular orders are identical. Yushmanov orders are used in Section 4 in a new approach of the problem of fitting of a tree metric to a given observed dissimilarity. Both Algorithms 4 and 5 of Section 4.1 involve the solution of a least squares approximation problem at each step. The property that Yushmanov orders are circular is extensively used to obtain the low time complexities of $O(n)$ in the reconstruction algorithm 2 of Section 3.1 and $O(n^2)$ in the fitting algorithm 5. The

presence of arbitrary initial choices in the last one deserves further studies. Presently, a procedure based on repeated uses of Algorithm 5 is proposed, involving all the possible first choices. In Section 4.2, some examples are presented. The procedure previously defined is compared with several good algorithms of the literature, and appears to be a competitive one. We conclude with some questions appealing for further developments.

2. The encoding of a valued tree with n leaves by $2n$-3 path lengths

2.1. Dissimilarities and trees. Let X be a set with n elements. A *dissimilarity* on X is a non-negative real function d on $X \times X$ satisfying the following two conditions:

- for all $x, y \in X$, $d(x,y) = d(y,x)$;
- for all $x, y \in X$, $d(x,y) \geq d(x,x) = 0$.

The dissimilarity d is a *metric* if it satisfies the classical metric triangular inequality:

for all $x, y, z \in X$, $d(x,z) \leq d(x,y) + d(y,z)$.

A graph G is also denoted as $(V(G),E(G))$, where $V(G)$ is the vertex set and $E(G)$ is a set of unordered pairs of distinct elements of $V(G)$, the edge set of G; for sake of brevity, an edge is denoted vv' instead of $\{v,v'\}$. The degree $\partial(v)$ of a vertex v is the number of edges $e \in E(G)$ such that $v \in e$. A *leaf* is a vertex of degree one. In a graph G, a *path* P between two vertices v and v' is a sequence of edges $vv_1, v_1v_2, \ldots, v_{k-1}v_k, v_kv'$. In fact, since only paths with all edges distinct will be considered here, a path will be generally identified with the set of its edges. A tree T is a graph with a unique path, denoted $T(uv)$, between any two distinct vertices u and v. A tree T has exactly $|V(T)|$-1 edges.

Let u be an inner vertex (that is, not a leaf) of a tree T, and an edge uv. Consider the set Y containing u and all the vertices v' of T such that $uv \in T(uv')$. The induced subtree T_Y is said to be a *branch of T at the vertex u*. All the leaves of a branch, except u, are leaves of T.

We mainly consider here the so-called X-trees, related with the set X by two properties: (i) the set of leaves of T is X; (ii) for any $v \in V(T)$-X, $\partial(v) \geq 3$. According to the terminology in the Barthélemy and Guénoche book [2], this means that the X-trees considered here are *separated* and *free*. In such an X-tree, the role of the inner vertices, called also *latent vertices* is just to determine the shape of the tree; they are often indicated without labels in figures. An X-tree has at most $2n$-2 vertices and thus $2n$-3 edges, these numbers corresponding to the non-degenerate case where all the latent vertices have degree 3. The *articulation vertex* $a(x)$ of $x \in X$ is the unique latent vertex adjacent to x (that is such that $xa(x) \in E(T)$). The number of edges of T is denoted as $\mu(T)$ (or simply μ).

We make correspond an X-tree T_ℓ to any valued tree $T'_{\ell'}$ with X as set of leaves by the repetition of the following operation: choose a vertex u of degree two in $V(T')$-X, delete it and replace the edges vu and uv' incident to u by a unique edge

vv'; set $\ell(vv') = \ell'(vu) + \ell'(uv')$. When no such vertex remains, an X-tree T is obtained; complete the end length function on T by setting $\ell(vv') = \ell'(vv')$ for all the edges common to T and T'. The trees T and T' have the same leaves and all the distances between leaves in $T'_{\ell'}$ are preserved in T_ℓ. We say that T_ℓ is the *reduced X-tree* corresponding with $T'_{\ell'}$.

We consider *valued X-trees* T_ℓ, where T is a tree and ℓ is a real *length* function on $E(T)$. The functions ℓ considered here are non-negative, with null values possible only on the edges adjacent to leaves. The *distance* $t(v,v')$ between two vertices v and v' is equal to $\sum_{e \in T(vv')} \ell(e)$; it satisfies the metric triangular inequality.

Let $\mathcal{X}$ be the class of all the X-trees T_ℓ, valued as above. A dissimilarity d on X is said to be a *tree metric* if it is representable by the length of the paths between the leaves of an element of $\mathcal{X}$. A tree metric is a metric. A dissimilarity on X satisfies the *four-point condition* if, for all $x, y, z, w \in X$,

$$d(x,y) + d(z,w) \leq \max\{\, d(x,z) + d(y,w)\,,\, d(x,w) + d(y,z)\,\}.$$

THEOREM 2.1 (Zaretskii [29], Buneman [5], Patrinos and Hakimi [22], Dobson [15]). *Let d be a dissimilarity on X. The following two conditions are equivalent:*
(i) *d is a tree metric;*
(ii) *d satisfies the four-point condition.*
Moreover, a tree metric admits a unique tree representation.

As recalled in Leclerc [17], a dissimilarity satisfying the four-point condition for all *distinct* $x, y, z, w \in X$ is not necessarily a metric, but is still uniquely representable by a valued X-tree, possibly with negative lengths on the edges adjacent to the leaves. Such a dissimilarity will be said a *tree dissimilarity*.

Given three distinct vertices u, v, w of a tree T, there is a unique vertex which is common to the three paths $T(uv)$, $T(vw)$ and $T(uw)$. This vertex is called the *median vertex* of the triple (u,v,w) and denoted as $m(u,v,w)$. If x, y, z are three leaves of T, then $m(x,y,z) \neq x, y, z$. In a valued tree T_ℓ, the distance between x and the path $T(yz)$ is equal to $t(x,m(x,y,z)) = \frac{1}{2}(t(x,y)+t(x,z)-t(y,z))$; this quantity is denoted as $dist(x,T(yz))$.

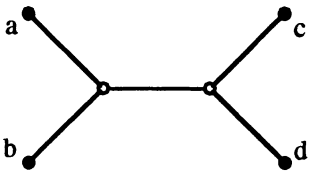


Figure 1

A triple (x,y,z) of leaves of T is said *well-formed* if $m(x,y,z) = a(y)$. Equivalently, the vertex $a(y)$ belongs to the path $T(xz)$. For instance, in the tree of

Figure 1, the triple (a,c,d) is well-formed, whereas (a,c,b) is not. Note that, if x and y are two leaves such that $a(x) = a(y)$, then every triple (x,y,z) is well-formed. Here is a simple characterization of well-formed triples in terms of distances:

PROPOSITION 2.2. *A triple (x,y,z) of leaves of T is well-formed if and only if the equality $dist(y,T(xz)) = \min_{w \in X,\, w \neq x,y} dist(y,T(xw))$ holds.*

Proof. If (x,y,z) is a well-formed triple, then, $dist(y,T(xz)) = \ell(ya(y))$. For every $w \in X$, $w \neq x, y$, consider the path $T(yv)$ from y to the vertex v which is the closest to y on the path $T(xw)$; the path $T(yv)$ includes the edge $ya(y)$, and, so, has a length at least equal to $\ell(ya(y))$. Conversely, the quantity $dist(y,T(xw))$ is minimized by those leaves w such that $a(y)$ belongs to the path $T(xw)$. ◊

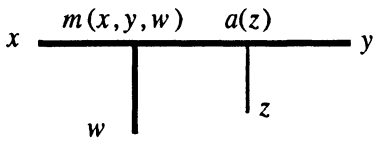


Figure 2

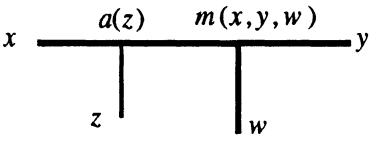


Figure 3

Bold lines represent paths ; thin ones represent single edges.

PROPOSITION 2.3. *Let d be a tree metric on the set X and x, y, z be three elements of X such that, in the X-tree representation T_ℓ of d, the triple (x,z,y) is well-formed. Then, the following equality holds for any $w \in X-\{x,y,z\}$:*

$$d(z,w) = \max\{\, d(x,w) + d(y,z)\,,\, d(y,w) + d(x,z)\,\} - d(x,y).$$

Proof. For each $w \in X-\{x,y,z\}$, the median vertex $m(x,y,w)$ lies either on the path $T(xa(z))$ (Figure 2), or on the path $T(ya(z))$ (Figure 3), or is equal to $a(z)$. The four-point condition expresses as $d(x,y) + d(z,w) = d(x,z) + d(y,w) \geq d(x,w) + d(y,z)$ in the first case, as $d(x,y) + d(z,w) = d(x,w) + d(y,z) \geq d(x,z) + d(y,w)$ in the second, and as $d(x,y) + d(z,w) = d(x,z) + d(y,w) = d(x,w) + d(y,z)$ in the degenerate third case. The result follows. ◊

So, when the triple (x,z,y) is well-formed, the distance between z and any element w of X may be computed from the values $d(x,y)$, $d(x,z)$, $d(y,z)$, $d(x,w)$ and $d(y,w)$.

The following notations are used when an indexing $x_1, x_2, \ldots, x_n$ of the set X is given. For three distinct leaves $x_i, x_j, x_k \in X$, we set $\Delta_{i,j}^k = dist(x_k,T(x_ix_j)) = \frac{1}{2}(d(x_i,x_k)+d(x_j,x_k)-d(x_i,x_j))$. Since d satisfies the triangular metric inequality, the quantities $\Delta_{i,j}^k$ are non negative: moreover, $d(x_i,x_j) = \Delta_{j,k}^i + \Delta_{i,k}^j$ and $\Delta_{i,j}^k > 0$ when the edge $x_ka(x_k)$ has a non-null length. We will also write a_k instead of $a(x_k)$.

2.2. Circular and diagonal orders. We now recall an encoding method, proposed in the Barthélemy and Guénoche [2] book as a generalization of a method given by Chaiken, Dewdney and Slater [7] in the special case of an unvalued tree. An order $x_1, x_2, \ldots, x_n$ on X is called a *diagonal order* of the X-tree T (Dewdney [14]) if, for any integer k (modulo n), the triple (x_{k-1}, x_k, x_{k+1}) is well-formed.

Consider a graphic planar representation of T (where two edges have no other common points than a vertex) and an order obtained as follows: first, the leaf x_1 is arbitrarily chosen; then, the leaves are indexed as $x_1, x_2, \ldots, x_n$ according to a circular (say, clockwise) scanning of the subset X of vertices of T. Such an order, frequently called *diagonal plane* in the literature, will be said here *circular*. It has the following property, for any integer k modulo n: when moving on the path $T(x_k x_{k+1})$ from x_k to x_{k+1}, all the branches at the encountered vertices are located on the right. A circular order is diagonal; otherwise, assume $m(x_{k-1}, x_k, x_{k+1}) \neq a(x_k)$: there is a branch of the tree at the vertex $a(x_k)$, the leaves of which being either between x_{k-1} and x_k or between x_k and x_{k+1} in the clockwise scanning, a contradiction with the hypothesis that the order is circular. Hence the following result holds:

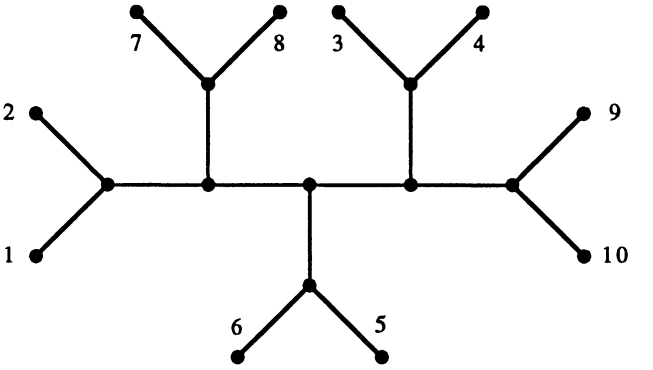


Figure 4

LEMMA 2.4. *Any X-tree T admits a diagonal order.*

As the following example shows, circular orders constitute in the general case a proper subclass of the diagonal ones. According to a remark above, if x and y are two leaves such that $a(x) = a(y)$, then the only condition on their places in a diagonal order is that they are consecutive (modulo n). So, in the example of Figure 4, the order (1 2 3 4 5 6 7 8 9 10) is diagonal. A simple investigation leads to the conclusion that it is not circular.

THEOREM 2.5 (Barthélemy and Guénoche [2]; see Leclerc and Makarenkov [18]). *Let* $X = \{ x_1, x_2, \ldots, x_n \}$ *be a linearly ordered set. For any sequence* $d_{12}, d_{13}, d_{23}, \ldots, d_{1i}, d_{i,i+1}, \ldots, d_{1n}, d_{n-1,n}$ *of* $2n-3$ *strictly positive real numbers, there exists a unique valued X-tree* T_ℓ *such that* $d_{ij} = \sum_{e \in T(x_i, x_j)} \ell(e)$ *and* $x_1, x_2, \ldots, x_n$ *is a circular order of* T.

As stated in Leclerc and Makarenkov [18], the function ℓ is non-negative if and only if the sequence $d_{12}, d_{13}, d_{23},\ldots, d_{1n}, d_{n-1,n}$ is extracted from a metric array. For a counter-example, consider the X-tree of Figure 5, where $X = \{x_1, x_2, x_3, x_4\}$. For such a tree, all the linear orders on X are circular. The sequence of five positive real numbers given by $d_{12} = d_{13} = d_{14} = d_{23} = 2$ and $d_{34} = 10$ leads to a system of five linear equations which has no solution for this tree: $\alpha+\beta = 2$; $\alpha+\gamma = 2$; $\alpha+\delta = 2$; $\beta+\gamma = 2$; $\gamma+\delta = 10$. In fact, the sequence corresponds to the valued tree of Figure 6. Since the sequence was not extracted from a metric array, this tree has a negatively valued edge and is the representation of a tree dissimilarity.

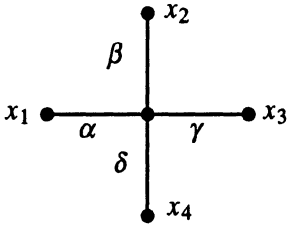


Figure 5

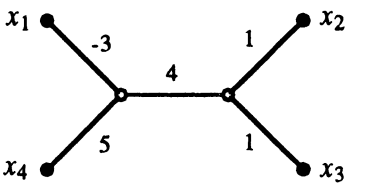


Figure 6

3. A combinatorial obtainment of circular orders

3.1. Yushmanov orders. In his 1984 paper [28], Yushmanov shows it possible to encode a positively valued X-tree T_ℓ by $2n$-3 lengths of paths between leaves. His work is independent of the Chaiken *et al.* one, even thought that, as these authors, he uses his results in a study of unvalued trees. The main results in Yushmanov's paper are recalled in this subsection with extensions to the cases of valued trees and dissimilarities. We also provide algorithms corresponding with Yushmanov's approach.

Let d be a tree metric on X, his matrix D, and consider the sets P of pairs of elements such that the knowledge of the entries $(d(x,y))$, $xy \in P$, allows us to recover the entire matrix D, provided the tree T is already known; the sets defined by Chaiken *et al.* and used in Theorem 2.5 are of this type. Denote as $\rho(T_\ell)$ the minimum cardinality of such a set. In his paper, Yushmanov first observes that every quantity $d(x,y)$ is a sum of lengths of edges of T. So, a subset P corresponds to a set of linear equations with rank $\mu(T) \leq 2n$-3. The equality $\rho(T_\ell) = \mu(T)$ follows; note that this observation remains valid when the edge lengths are no longer assumed to be positive. Similarly, consider the sets Q such that the knowledge of the entries $(d(x,y))$, $xy \in Q$, allows us to recover D without any further requirements. Denote their minimum cardinality as $\delta(T_\ell)$. Obviously, $\rho(T_\ell) \leq \delta(T_\ell)$. Indeed, Leclerc [17] gives two ways of determining a set Q of cardinality $2n$-3 = $\rho(T_\ell)$, thus proving the equality $\rho(T_\ell) = \delta(T_\ell)$. Yushmanov was the first to exhibit such sets, related with linear orderings $x_1, x_2,\ldots, x_n$ of X such that, for all $k = n, n$-1,..., 2, the triple (x_1,x_k,x_{k-1}) is well-formed in a current tree T^k with k leaves. Such an order, called hereafter a *Yushmanov order*, is obtained by the procedure 1 described below:

Procedure 1

Initialization. Choose arbitrarily two leaves of T_ℓ and index them as x_1 and x_n. Set $V^n = \emptyset$ and $T^n = T$.

Step 1. Choose a leaf x_{n-1} in T^n in such a way that the vertex a_n lies on the path $T^n(x_1x_{n-1})$. Delete the leaf x_n and the edge a_nx_n in T^n and reduce the resulting tree in order to obtain the tree T^{n-1}; set $V^{n-1} = \{x_n\}$.

Step k+1. Assume the first k steps have led to a tree T^{n-k} the set of leaves of which is X-V^{n-k}, where $V^{n-k} = \{x_n,\ldots,x_{n-k+1}\}$.

If $k = n$-2, then $V^2 = \{x_n,\ldots,x_3\}$ and T^2 is reduced to the unique edge x_1x_2; since x_1 is already fixed, the Yushmanov indexing $x_1, x_2,\ldots, x_n$ is completely determined.

Otherwise, choose a leaf x_{n-k-1} in T^{n-k} in such a way that the vertex a_{n-k} lies on the path $T^{n-k}(x_1,x_{n-k-1})$; delete the leaf x_{n-k} and the edge $a_{n-k}x_{n-k}$ in T^{n-k}; reduce the resulting tree in order to obtain the tree T^{n-k-1}; set $V^{n-k-1} = V^{n-k}\cup\{x_{n-k}\}$.

The choice of the leaf x_{n-k-1} is always possible: the vertex a_{n-k} adjacent to x_{n-k} in T^{n-k} has degree at least three, defining at least three branches. Any leaf which belongs to a branch which neither contains the leaf x_1 nor consists of the edge $a_{n-k}x_{n-k}$ may be chosen as x_{n-k-1}.

The remaining problem is the determination of a Yushmanov order without the knowledge of the tree T. A solution is provided by arguments already used in the proof of Proposition 2.2: the possible choices are the elements x_{n-k-1} (different from x_1 and x_{n-k}) such that $dist(x_{n-k},T(x_{n-k-1}x_1)) = \min_{w\in X-V^{n-k}} dist(x_{n-k},T(wx_1))$. With the expression of $dist(x_{n-k},T(wx_1))$ recalled in Section 2.1, x_{n-k-1} is an element $w \in X$-V^{n-k} minimizing the difference $d(x_{n-k},w)$-$d(x_1,w)$. Such an element is chosen directly on the matrix of d in the formal statement of Algorithm 1 below (see the notations of Section 2.1). In fact, Algorithm 1 works with any dissimilarity matrix, tree metric or not, as input; so, the definition of a Yushmanov order of X extends to all dissimilarities.

Algorithm 1: construction of a Yushmanov indexing $x_1, x_2,\ldots, x_n$ of the set X:

Input: a finite set X with n elements ; a dissimilarity d on X.

Output: a Yushmanov order $(x_1, x_2,\ldots, x_n)$ on X associated with d.

Initialization Choose arbitrarily two leaves x_1 and x_n; $W := X$-$\{x_1,x_n\}$; $k = 0$

Repeat

Find x_{n-k-1} in W such that:

$d(x_{n-k},x_{n-k-1})$-$d(x_1,x_{n-k-1}) = \min_{w\in W} d(x_{n-k},w)$-$d(x_1,w)$;

$W := W$-$\{x_{n-k}\}$;

$k = k$+1

Until $W = \emptyset$

In the k-th step of Algorithm 1, n-k-2 elements of X are examined. So, this algorithm has time complexity $O(n^2)$. A converse procedure allows to reconstruct the valued X-tree T_ℓ from the tree metric d and a Yushmanov order $x_1, x_2,\ldots, x_n$

obtained by Algorithm 1 applied to d. Consider the sequence $(d(x_1,x_2), d(x_1,x_3), d(x_2,x_3),\ldots, d(x_1,x_i), d(x_i,x_{i+1}),\ldots, d(x_1,x_n), d(x_{n-1},x_n))$ with $2n$-3 terms. For $k = 2,\ldots, n$-1, we are able to compute the quantities $\Delta^1_{k,k+1} = dist(x_1,T(x_kx_{k+1})) = \frac{1}{2}(d(x_1,x_k) + d(x_1,x_{k+1}) - d(x_k,x_{k+1}))$ and $\Delta^{k+1}_{1,k} = dist(x_{k+1},T(x_1x_k)) = \frac{1}{2}(d(x_1,x_{k+1}) + d(x_k,x_{k+1}) - d(x_1,x_k))$; they are assumed to be positive, as it is the case when the values d_{ij} are extracted from a distance matrix. A sequence of valued trees T^k_ℓ with k leaves, $k = 2,\ldots, n$, is constructed according to the following Procedure 2:

Procedure 2

Step 1 (initialization). T^2_ℓ is the tree reduced to the unique edge x_1x_2 with length $\ell(x_1x_2) = d(x_1,x_2)$.

Step k ($k = 2,\ldots,n$-1). A tree T^k_ℓ with the leaves $x_1, x_2,\ldots, x_k$ has been built. Two cases may occur for the path $T^k_\ell(x_1x_k)$:

Case 1. There exists a vertex u on this path such that $t(x_1,u) = \Delta^1_{k,k+1}$. In this case, the leaf x_{k+1} is the only vertex to add to T^k_ℓ in order to obtain T^{k+1}_ℓ, with the new edge ux_{k+1} of length $\ell(ux_{k+1}) = \Delta^{k+1}_{1,k}$.

Case 2. There exists an edge vv' on the path $T^k_\ell(x_1,x_k)$ such that $t(x_1,v) < \Delta^1_{k,k+1} < t(x_1,v')$. In this case, a new inner vertex u is added on the edge vv', now divided into two edges uv and uv', with lengths $\ell(uv) = \Delta^1_{k,k+1} - t(x_1,v)$ and $\ell(uv') = t(x_1,v') - \Delta^1_{k,k+1}$; then, as before, the leaf x_{k+1} is added to T^k_ℓ in order to obtain T^{k+1}_ℓ, with the new edge ux_{k+1} of length $\ell(u,x_{k+1}) = \Delta^{k+1}_{1,k}$.

For $k = n$, the valued tree T^n_ℓ is equal to T_ℓ.

In the next Algorithm 2, the k-th step consists of the examination from edge to edge (starting from x_k) of the path $T(x_kx_1)$ until the good place for the articulation vertex a_{k+1}, depending on $\Delta^k_{1,k+1}$, is found. The new leaf x_{k+1} is then added to the tree T with a new edge $a_{k+1}x_{k+1}$ of length $\Delta^{k+1}_{1,k}$. Note that the number of edges examined at this step is no more than $|T^n(x_kx_{k+1})|$, the number of edges of the path between x_k and x_{k+1} in the final tree T^n. Some edges are recognized as not belonging to a current path $P(T)$ in the next step k+1 and in the sequel; such edges are included in the set $E(T)$ (and their extremities in $V(T)$). This is due to the observation that any edge excluded from the linked list $P(T)$ will never return in this list, since $P(T)$ is always completed with one or two new edges. The complexity of Algorithm 2 is presently estimated as $O(\sum_{1\le k\le n-1} |T^n(x_kx_{k+1})|) + O(n)$, and will be shown to be in fact $O(n)$ in Section 3.4 (Corollary 3.7).

Some further notations are used in the following formal statement of Algorithm 2: given two leaves x, y of an X-tree T, $\ell(i,T(xy))$, $w(i,T(xy))$ and $w'(i,T(xy)) = w(i+1,T(xy))$ are respectively the length, the initial vertex and the terminal vertex of the i–th edge (starting from x) of the path $T(xy)$; $P(T)$ is the linked list of the edges of the path $T(x_1x_k)$ (starting from x_1), and $E(P(T))$ and $V(P(T))$ are, respectively, the edge set and the vertex set of $P(T)$.

Algorithm 2: Reconstructing a valued tree T_ℓ from a tree metric d and a corresponding Yushmanov order on X.

Input: a finite set X with n elements; the $2n$-3 entries $d(x,y)$, $xy \in \{x_1x_2, x_1x_3, x_2x_3, \ldots, x_1x_i, x_ix_{i+1}, \ldots, x_1x_n, x_{n-1}x_n\}$ corresponding with a Yushmanov order $(x_1, x_2, \ldots, x_n)$ for a tree metric d on X.

Output: the valued X-tree $T_\ell = (V(T),E(T),\ell)$ associated with d.

Initialization $V(T) := \emptyset$; $E(T) := \emptyset$; $k := 1$; $P(T) := \{x_1x_2\}$;
$\ell(x_1x_2) := d(x_1,x_2)$

Repeat
$k := k+1$; $i := 1$; $S := 0$
if $\Delta^k_{1,k+1} > 0$ **then**
$S := \ell(1,T(x_kx_1))$
$u := w'(1,T(x_kx_1))$
$v := w(1,T(x_kx_1)) = x_k$
$P(T) := P(T) - \{uv\}$
if $S \geq \Delta^k_{1,k+1}$ **then**
$V(T) := V(T)+\{x_k\}$
else $u := x_k$
while $S < \Delta^k_{1,k+1}$ **do**
$V(T) := V(T)+\{u,v\}$
$E(T) := E(T)+\{uv\}$
$i := i+1$
$u := w'(i,T(x_kx_1))$
$v := w(i,T(x_kx_1))$
$S := S+\ell(i,T(x_kx_1))$
$P(T) := P(T) - \{uv\}$
if $S > \Delta^k_{1,k+1}$ **then**
$E(T) := E(T)+\{a_{k+1}v\}$
$P(T) := P(T)+\{ua_{k+1}\}+\{a_{k+1}x_{k+1}\}$
$\ell(a_{k+1}v) := \Delta^k_{1,k+1}-S+\ell(uv)$
$\ell(ua_{k+1}) := S-\Delta^k_{1,k+1}$
$\ell(a_{k+1}x_{k+1}) := \Delta^{k+1}_{1,k}$
else $P(T) := P(T)+\{ux_{k+1}\}$
$\ell(ux_{k+1}) := \Delta^{k+1}_{1,k}$
until $(k = n\text{-}1)$
$E(T) := E(T)+E(P(T))$; $V(T) := V(T)+V(P(T))$

THEOREM 3.1 (Yushmanov [28]). *The successive uses of Procedures 1 and 2 map any valued X-tree T_ℓ on itself.*

Proof. The result is true for $n = 3$: in this case, the X-tree T has one latent vertex a adjacent to its three leaves and every order on X is Yushmanov. For an arbitrary order x_1, x_2, x_3, the lengths $\Delta^1_{2,3}$, $\Delta^2_{1,3}$ and $\Delta^3_{1,2}$ of the edges ax_1, ax_2 and ax_3 are determined by Procedure 2.

Assume that the result is true for every X'-tree, where X' is a set of cardinality n-1. Then, let $x_1,\ldots, x_n$ be a Yushmanov order on X. By the induction hypothesis, Procedure 2 constructs, before its last iteration, a valued tree T_ℓ^{n-1} with n-1 leaves such that: (i) the leaves of T_ℓ^{n-1} are the elements $x_1, x_2,\ldots, x_{n-1}$; and (ii) the distances between these leaves are given by the restriction of d to X-$\{x_n\}$. By the rule used for choosing x_{n-1} at the first iteration of Procedure 1, one also knows that the latent vertex a_n lies on the path $T^{n-1}(x_1x_{n-1})$; then, the determination of the place of a_n on this path is made in the last step of Procedure 2 on such a way that the distance d is realized by the tree T_ℓ^n for the pairs x_1x_n and x_nx_{n-1}. By the proposition 2.3, the induction hypothesis, and the unicity of the tree representation recalled in Theorem 2.1, $T_\ell^n = T_\ell$ is the unique valued X-tree realizing the distance d on X. ◊

Many fitting algorithms of the literature transform a given dissimilarity matrix d_0 on X into a tree metric one d; this is the case of the decomposition algorithm of Brossier [4], the algorithm based on minimum spanning trees of Leclerc [17], or the reduction methods of Roux [25] and Gascuel and Levy [16]. Then, the problem of the reconstruction of the X-tree representation of d remains. Methods like ADDTREE (Sattah and Tversky [27]) or the scoring method of Luong [19] and Barthélemy and Guénoche [2] are sometimes proposed in the literature for determining the corresponding X-tree; as their time complexity $O(n^5)$ indicates, these methods have not been designed for this particular use. Starting from the distance matrix of d, the successive uses of Algorithms 1 and 2 provide the valued X-tree in $O(n^2)$ time, which does not exceed the complexity of any fitting method.

3.2. An example. For an illustration of Algorithms 1 and 2, consider the valued X-tree of Figure 7. Let us start from the corresponding tree metric array (Table 1).

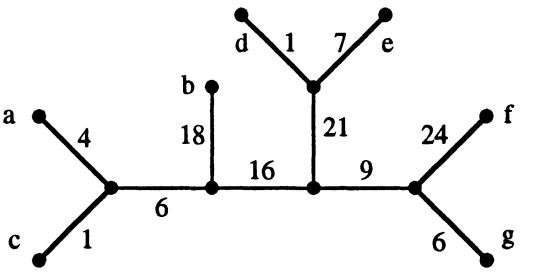


Figure 7

Set x_1 = a and x_7 = b ; then, Algorithm 1 computes the quantities $dist$(b,T(ax)) for x = c , d , e , f , g , which gives:

$2dist$(b,T(ac)) = 28+25-5 = 48
$2dist$(b,T(ad)) = 28+56-48 = 36
$2dist$(b,T(ae)) = 28+62-54 = 36
$2dist$(b,T(af)) = 28+67-59 = 36
$2dist$(b,T(ag)) = 28+49-41 = 36

So, x_6 may be chosen among d , e , f and g ; set, for instance, x_6 = d and compute:

$2dist$(d,T(ac)) = 48+45-5 = 88
$2dist$(d,T(ae)) = 48+8-54 = 2

$2dist(d,T(af)) = 48+55-59 = 44$ $2dist(d,T(ag)) = 48+37-41 = 44$
So, $x_5 =$ e and, at the next step, we have: $2dist(e,T(ac)) = 54+51-5 = 100$
$2dist(e,T(af)) = 54+61-59 = 56$ $2dist(e,T(ag)) = 54+43-41 = 56$
Choosing $x_4 =$ f, we compute: $2dist(f,T(ac)) = 59+56-5 = 110$
$2dist(f,T(ag)) = 59+30-41 = 48$, leading to $x_3 =$ g and $x_2 =$ c.

b	28					
c	5	25				
d	48	56	45			
e	54	62	51	8		
f	59	67	56	55	61	
g	41	49	38	37	43	30
	a	b	c	d	e	f

Table 1

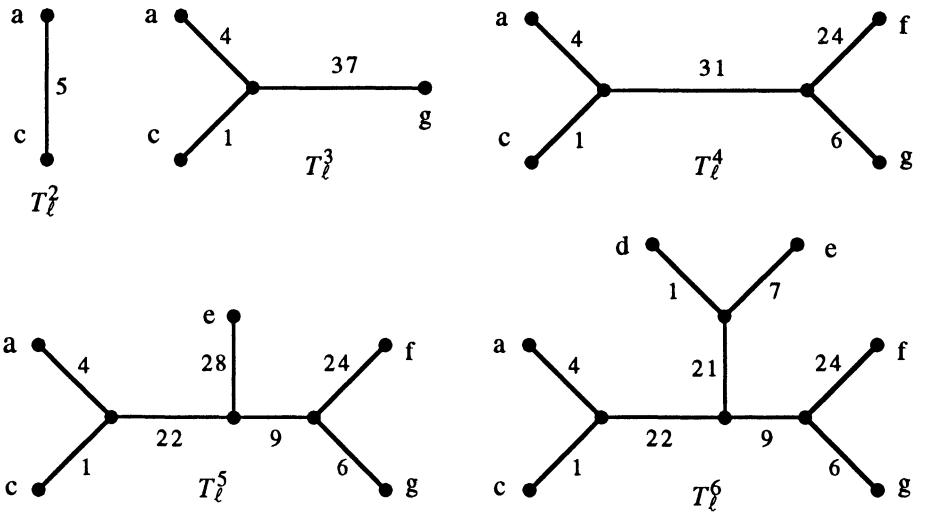


Figure 8

3.3. Yushmanov and circular orders are the same. As noticed just above, circular and Yushmanov orders are defined on two different ways. Both ways select well-formed triples. According to the results of this section, these two ways lead in fact to the same orders.

PROPOSITION 3.2. *Every circular order on the leaves of an X-tree T is Yushmanov.*

Proof. We prove the result by induction on n; for $n = 3$, all orders on X are both circular and Yushmanov.

Assume the result is true for all X-trees with at most n-1 leaves. Let T be an X-tree with n leaves $x_1, x_2,\ldots, x_n$, indexed accordingly with a circular order. Consider the path $T(x_1x_{n-1})$ between x_1 and x_{n-1}. Since the triple (x_{n-1},x_n,x_1) is well-formed, we have the configuration of Figure 9, with $m(x_{n-1},x_n,x_1) = a_n$. So, the vertex a_n belongs to the path $T(x_1x_{n-1})$; it follows that Procedure 1 of Section 3.1 can be performed to obtain a Yushmanov indexing $y_1, y_2,\ldots, y_n$ of X, with choices in Initialization and Step 1 leading to $y_1 = x_1$, $y_{n-1} = x_{n-1}$ and $y_n = x_n$.

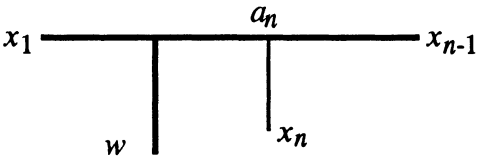


Figure 9 (with the same conventions as Figures 2-3)

Then, in the tree T, delete the vertex x_n and the edge a_nx_n and reduce the obtained tree. A new tree T' with the leaves $x_1,\ldots, x_{n-1}$ is obtained. From a planar representation of T admitting the previous circular indexing $x_1, x_2,\ldots,x_n$, we derive a planar representation of T' with $x_1,\ldots, x_{n-1}$ as a circular indexing. So, by the induction hypothesis, this order is Yushmanov. It follows that it can be obtained by Procedure 1, with x_1 and x_{n-1} as vertices chosen in the initial step; that is, the Initialization and Step 1 mentioned just above can be continued according to Procedure 1 in order to obtain the order $x_1,\ldots, x_n$ on X. ◊

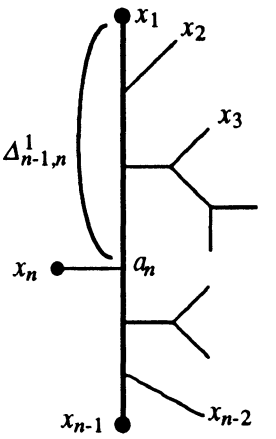


Figure 10

PROPOSITION 3.3. *Every Yushmanov order on the leaves of an X-tree T is circular.*

Proof. Let T be an X-tree with n leaves $x_1, x_2,\ldots, x_n$, indexed according to a Yushmanov order. We must show that there exists a planar drawing of T for which this order is circular. This is obviously true for $n = 3$, and we proceed again by induction on n.

Assume the result is true for all X-trees with at most n-1 leaves, and let T be an X-tree with n leaves. After deletion of the edge $a_n x_n$ and reduction of the obtained tree, a new tree T' is obtained, with leaves $x_1,\ldots, x_{n-1}$ still indexed accordingly with a Yushmanov order. By the induction hypothesis, there exists a planar drawing of T' such that this order is circular. As a consequence, there is no branch on the left when moving from x_{n-1} to x_1 on the path $T(x_{n-1}x_1)$. Add the vertex a_n on this path at the place specified by Algorithm 2; the drawing of the edge $a_n x_n$ on the left of the path $T(x_{n-1}x_1)$ gives $x_1, x_2,\ldots, x_n$ as a circular order of T (Figure 10). ◊

Propositions 3.2 and 3.3 assemble into the following

THEOREM 3.4. *An order $x_1, x_2,\ldots, x_n$ on the leaves of an X-tree T is circular if and only if it is Yushmanov.*

Now, Yushmanov's algorithm 1 appears to be a purely combinatorial, free of geometric representation, construction of circular orders. For a dissimilarity d on X, a linear order C on X derived from d by Algorithm 1 will be called "circular" if d is known to be a tree metric, and "Yushmanov" otherwise.

3.4. Further properties of circular orders and recognition of a tree metric. The following Algorithm 3 decides whether a given dissimilarity matrix d is a tree metric. The elements of X are examined in an order provided by Algorithm 1. When x_{k+1} is examined, Proposition 2.3 gives a formula leading to values $t(j,k+1)$ of the distance between the leaves x_j and x_{k+1} in the tree T, for all $j = 2,\ldots, k$-1, under the hypothesis that d is a tree metric. These computed values are compared with the actual ones and the conclusion follows; in the statement below, the "return" command means the exit from the algorithm.

Algorithm 3: recognition of a tree metric

Input: a finite set X with n elements; a dissimilarity d on X and a corresponding Yushmanov order $(x_1, x_2,\ldots, x_n)$ on X.

Output: the answer "Yes" if d is a tree metric; the answer "No" otherwise.

Initialization
Compute $\Delta^1_{2,3}$, $\Delta^2_{1,3}$ and $\Delta^3_{1,2}$
if ($\Delta^1_{2,3} \geq 0$, $\Delta^2_{1,3} \geq 0$, $\Delta^3_{1,2} \geq 0$)
else print "No"
return
$k := 3$
repeat
Compute $\Delta^1_{k,k+1}$, $\Delta^k_{1,k+1}$ and $\Delta^{k+1}_{1,k}$

```
if    (Δ^1_{k,k+1} ≥ 0, Δ^k_{1,k+1} ≥ 0, Δ^{k+1}_{1,k} ≥ 0) then
      for  j = 2,..., k-1  do
           t(j,k+1) := max{d(x1,xj)+d(xk,xk+1),d(x1,xk+1)+d(xj,xk)}-d(x1,xk)
           if   (t(j,k+1) ≠ d(xj,xk+1)) then
                print "No"
                return
      else print "No"
           return
      k := k+1
until (k = n+1)
print "Yes"
```

Recall $\Delta^k_{ij} = dist(x_k, T(x_i x_j)) = \frac{1}{2}(d(x_i,x_k)+d(x_j,x_k)-d(x_i,x_j))$. The time complexity of Algorithm 3 is $O(n^2)$, similar to previous algorithms addressing the same problem (final note in Bandelt [1], Leclerc [17]).

Theorem 3.4 suggests some remarks about circular orders. First, as illustrated in the example of Section 3.2 above, when d is a tree metric, several Yushmanov orders may be obtained with the same initial vertices x_1 and x_n; this is due to the fact that tree metrics are strongly constrained, although their regularities do not directly appear in their matrices. So, arbitrary choices may be needed at every step. This explains why the number $n(n\text{-}1)$ of possible initial choices differs from the number of circular orders (defined modulo n), for instance 2^{n-2} for an X-tree with the maximum number $n\text{-}2$ of latent vertices. On the other hand, in the general case of a dissimilarity d without special properties, ties on the values of $d(x_{n-k},w)-d(x_1,w)$ rarely occur in Algorithm 1 and the number of different Yushmanov orders is close to $n(n\text{-}1)$. Note also that the choice of the initial vertex x_1 in Algorithm 1 finally does not matter when the purpose is just to obtain a circular order, because of the following consequence of Theorem 3.4:

COROLLARY 3.5. *If $x_1, x_2, \ldots, x_n$ is a circular order on the leaves of an X-tree T, then, for any $k \in \{1,\ldots,n\}$, the order $x_k, x_{k+1}, \ldots, x_n, x_1, x_2, \ldots, x_{k-1}$ is again circular.*

Another property of circular orders is related with the induction on the number n used in the proof of Proposition 3.3. With the remark that the difference between $d(x_{n-1},x_n)+d(x_n,x_1)$ and $d(x_{n-1},x_1)$ is two times the length of the new edge $a_n x_n$, the following Proposition 3.6, already stated by Yushmanov, holds:

PROPOSITION 3.6. *The sum $\ell(T)$ of the lengths of the edges of a valued X-tree T_ℓ is given by $2\ell(T) = d(x_1,x_2) + d(x_2,x_3) + d(x_3,x_4) + \ldots + d(x_{n-1},x_n) + d(x_n,x_1)$, provided $x_1, x_2, \ldots, x_n$ is a circular order on X.*

Especially, in the unvalued case, the sum $d(x_1,x_2) + d(x_2,x_3) + \ldots + d(x_n,x_1)$ is two times the number of edges of the tree.

COROLLARY 3.7. *Algorithm 2 reconstructs a valued X-tree T with n leaves in $O(n)$ operations.*

Proof. As a consequence of the previous result applied to the unvalued case, the number $\sum_{1\leq k\leq n-1} |T^n(x_k x_{k+1})|$, which is an upper bound of the number of the edges examined in the algorithm, is two times the number of edges of the final tree. ◊

4. A fitting method

4.1. Two fitting algorithms. This section is devoted to the problem, very often addressed in the literature, of fitting a tree metric t to a given dissimilarity d. Algorithms of various types have been given or recalled, for instance, by De Soete [12], Luong [19], Saitou and Nei [26], Barthélemy and Guénoche [2], Roux [25], Leclerc [17], De Soete and Caroll [13], Gascuel and Lévy [16]. The best time complexity of such algorithms is $O(n^2)$; the algorithms reaching such a complexity are rarely good for global criteria like the least squares one. In fact, the least square approximation of a dissimilarity d by a tree metric t is shown to be NP-hard in Day [10] (see also Day [11]). For this problem, the heuristics giving satisfactory results have usually a time complexity of $O(n^4)$ or $O(n^5)$, the Saitou and Nei NJ (nearest joining) method in $O(n^3)$ being a noticeable exception. For many problems of data analysis, where the purpose is to handle large data sets, a complexity order of $O(n^4)$ or $O(n^5)$ is too high. Since $O(n^5)$ algorithms are still currently proposed for the fitting of tree metrics, it seems that the situation is different in many applications of this problem. We describe here two algorithms based on Yushmanov orders. They proceed by successive local least squares approximations and are basically in $O(n^2)$. Global approximation and repetitive uses will increase this complexity up to $O(n^4)$ or $O(n^5)$ in Section 4.2, with seemingly good performances (Section 4.3).

The principle of the algorithm is as follows: at the step k, $2 \leq k \leq n$-1, a current valued tree T_ℓ^k has been determined, with the leaves $\{x_1,\ldots, x_k\}$. The vertex a_{k+1} is assumed to be on the path $T_\ell^k(x_1x_k)$ of this tree and a reconstruction procedure is introduced to obtain the tree T_ℓ^{k+1}, with the further problem of the determination of the best place of a_{k+1} on the path $T_\ell^k(x_1x_k)$. At the final step n-1, the valued X-tree T_ℓ corresponding to the tree metric t is obtained. In the determination of the best place of a_{k+1}, the lengths α, β and γ, of, respectively, the paths $T_\ell^{k+1}(x_1a_{k+1})$ and $T_\ell^{k+1}(a_{k+1}x_k)$ and the edge $a_{k+1}x_{k+1}$, are adjusted at each step according to a least squares criterion. Two methods are proposed here. In Algorithm 4, the computations at the Step k are based on the only two values $d(x_1,x_{k+1})$, $d(x_k,x_{k+1})$ of the initial dissimilarity, together with the value $t(x_1,x_k)$ determined at the previous step; this computation corresponds with Problem $P_{1,k}$. In Algorithm 5, the best place for a_{k+1} is determined for each edge uv of the path $T_\ell^k(x_1x_k)$, taking in account all the initial values $d(x_i,x_{k+1})$, $i = 1,\ldots,k$. This computation corresponds with Problem $P_{2,k}(uv)$. The edge leading to the best fitting is chosen.

Problem $P_{1,k}$ (see Figure 11):

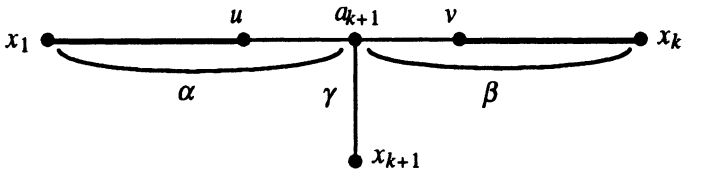


Figure 11

MINIMIZE $(\alpha + \gamma - d(x_1,x_{k+1}))^2 + (\beta + \gamma - d(x_k,x_{k+1}))^2$,
subject to: $\alpha+\beta = t(x_1,x_k)$ (determined at the previous step); $\alpha \geq 0$; $\beta \geq 0$; $\gamma \geq 0$.

Set $A = d(x_k,x_{k+1}) - t(x_1,x_k) - d(x_1,x_{k+1})$ and $B = t(x_1,x_k) - d(x_k,x_{k+1}) - d(x_1,x_{k+1})$; the problem reduces as:

MINIMIZE $\alpha^2 + \gamma^2 + A\alpha + B\gamma$,
subject to: $\alpha \geq 0$; $\gamma \geq 0$; $t(x_1,x_k) - \alpha \geq 0$.

With the use of the Lagrange function (see for instance Ciarlet [8] or Minoux [21]):

$$F(\lambda_1,\lambda_2,\lambda_3) = \alpha^2 + \gamma^2 + A\alpha + B\gamma - \lambda_1\alpha - \lambda_2\gamma + \lambda_3(\alpha - t(x_1,x_k)),$$

where $\lambda_i \geq 0$ for $i = 1, 2, 3$, we obtain a necessary condition on α and γ for reaching the minimum: $(\alpha,\gamma) \in \{(-\frac{A}{2},-\frac{B}{2}); (-\frac{A}{2},0); (t(x_1,x_k),-\frac{B}{2}); (t(x_1,x_k),0); (0,-\frac{B}{2}); (0,0)\}$.

Among these couples, choose the one satisfying the constraints and actually realizing the minimum.

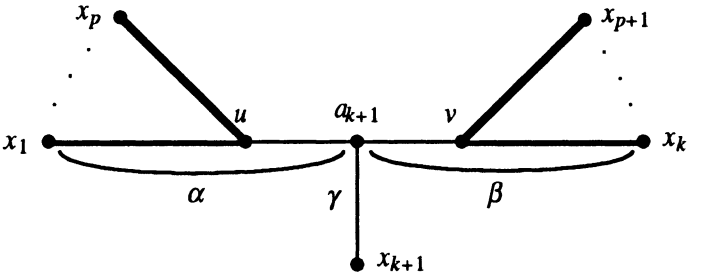


Figure 12

Problem $P_{2,k}$:
Let uv be an edge of the path $T_\ell^k(x_1x_k)$, u being its extremity closest to x_1. Since the order $x_1, x_2, \ldots, x_k$ on the leaves of T_ℓ^k is circular, there always exists an index p such that all the leaves $x_1, \ldots, x_p$ are on the same side of a_{k+1}, while $x_{p+1}, \ldots, x_k$ are on the other side (Figure 12). Taking in account the distances $dist(x_i,T(x_1x_k))$ for all elements x_i, $2 \leq i \leq k\text{-}1$, we obtain the following quantity to minimize for the best place for a_{k+1} on the edge uv:

Problem $P_{2,k}(uv)$

MINIMIZE $(\alpha + \gamma - d(x_1,x_{k+1}))^2 + (\beta + \gamma - d(x_k,x_{k+1}))^2$
$+ \sum_{2\le i\le p} (d(x_i,x_{k+1}) - (\alpha - dist(x_1,T(x_ix_k)) + dist(x_i,T(x_1x_k)) + \gamma))^2$
$+ \sum_{p+1\le i\le k-1} (d(x_i,x_{k+1}) - \beta - dist(x_k,T(x_1x_i)) + dist(x_i,T(x_1x_k)) + \gamma)^2,$
subject to: $\alpha + \beta = t(x_1,x_k)$; $\beta \ge 0$; $\gamma \ge 0$; $t(x_1,u) \le \alpha \le t(x_1,v)$,

where $t(x_1,u)$ and $t(v,x_k)$ are the distances between the corresponding vertices in the valued tree T_ℓ^k.

Set $A_1 = d(x_1,x_{k+1})$, $A_k = d(x_k,x_{k+1}) - t(x_1,x_k)$,
$A_i = d(x_i,x_{k+1}) - t(x_i,x_k) + t(x_1,x_k)$ for $2 \le i \le p$, and
$A_i = d(x_i,x_{k+1}) - t(x_1,x_i)$ for $p+1 \le i \le k-1$. After reduction, the problem is now:

MINIMIZE $\sum_{1\le i\le p} (\alpha+\gamma-A_i)^2 + \sum_{p+1\le i\le k} (\alpha-\gamma+A_i)^2,$
subject to: $\gamma \ge 0$; $t(x_1,u) \le \alpha \le t(x_1,v)$.

Setting $B = 4p-2k$, $C = 2\sum_{p+1\le i\le k} A_i - 2\sum_{1\le i\le p} A_i$ and $D = -2\sum_{1\le i\le k} A_i$, one gets:

MINIMIZE $k\alpha^2 + k\gamma^2 + B\alpha\gamma + C\alpha + D\gamma,$
subject to the same constraints.

Consider the Lagrange function:

$$F(\lambda_1,\lambda_2,\lambda_3) = k\alpha^2+k\gamma^2+B\alpha\gamma+C\alpha+D\gamma+\lambda_1(\alpha-t(x_1,v))-\lambda_2\gamma+\lambda_3(t(x_1,u)-\alpha).$$

The necessary conditions for minimum are:

$F'_\alpha = 2k\alpha + B\gamma + C + \lambda_1 - \lambda_3 = 0;$
$F'_\gamma = 2k\gamma + B\alpha + D - \lambda_2 = 0;$
$\lambda_1(\alpha-t(x_1,v)) = 0$; $\lambda_2\gamma = 0$; $\lambda_3(t(x_1,u)-\alpha) = 0,$

where $\lambda_j \ge 0$ for $j = 1, 2, 3$.

This system of equations leads to six possible solutions:

1. $\alpha = t(x_1,v)$, $\gamma = 0$;
2. $\alpha = t(x_1,v)$, $\gamma = -\dfrac{B(t(x_1,v)+D}{2k}$;
3. $\alpha = -\dfrac{C}{2k}$, $\gamma = 0$;
4. $\alpha = \dfrac{BD-2kC}{4k^2-B^2}$, $\gamma = \dfrac{BC-2kD}{4k^2-B^2}$;
5. $\alpha = t(x_1,u)$, $\gamma = 0$;
6. $\alpha = t(x_1,u)$, $\gamma = -\dfrac{B(t(x_1,u)+D}{2k}$.

Among the couples (α,γ) above, choose the one satisfying the constraints and actually realizing the minimum. End of Problem $P_{2,k}(uv)$.

Among the edges of the path $T_\ell^k(x_1x_k)$, choose the one realizing the minimum. End of Problem $P_{2,k}$.

In the following statement, the notations are the same as in Algorithm 2; moreover, $w(0,T(x_1x_k)) = w(1,T(x_1x_k)) = w'(0,T(x_1x_k)) = x_1$.

Algorithm 4: construction of a valued X-tree from a dissimilarity d

Input: a finite set X with n elements; a dissimilarity d on X.

Output: a valued X-tree $T_\ell = (V(T),E(T),\ell)$.

Initialization. Compute a Yushmanov order $(x_1, x_2,..., x_n)$ on X;
$V(T) := \{x_1, x_2\}$; $E(T) := x_1x_2$; $\ell(x_1x_2) := d(x_1,x_2)$; $k := 1$
Repeat
$k := k+1$; $S := 0$; $i := 0$
The problem is to add the leaf x_{k+1} to the current valued tree T_ℓ^k with leaves $x_1,..., x_k$.
Solve Problem $P_{1,k}$ (α, γ) for the path $T_\ell^k(x_1x_k)$
while $S < \alpha$ **do**
$i := i+1$
$S = S+\ell(i,T(x_1x_k))$
$u := w(i,T(x_1x_k))$; $v := w'(i,T(x_1x_k))$
if $S = \alpha$ **then**
$V(T) := V(T)\cup\{x_{k+1}\}$
$E(T) := E(T)\cup\{ux_{k+1}\}$
$\ell(ux_{k+1}) := \gamma$
else $V(T) := V(T)\cup\{a_{k+1},x_{k+1}\}$
$E(T) := (E(T)-\{uv\})\cup\{ua_{k+1},va_{k+1},a_{k+1}x_{k+1}\}$
$\ell(a_{k+1}x_{k+1}) := \gamma$
$\ell(ua_{k+1}) := \alpha - S + \ell(i,T(x_1x_k))$
$\ell(va_{k+1}) := S - \alpha$
until $(k = n)$

Algorithm 5: construction of a valued X-tree from a dissimilarity d

This algorithm is identical to Algorithm 4, except the instruction "**Solve** Problem $P_{1,k}$", which is replaced with "**Solve** Problem $P_{2,k}$".

4.2. Time complexity and strategies for the use of Algorithms 4 and 5. Problem $P_{1,k}$ is solved in $O(1)$ and the obtainment of a Yushmanov order by Algorithm 1 is $O(n^2)$. Then, Algorithm 4 has the same time complexity $O(n^2)$. In Problem $P_{2,k}$, using the fact that the Yushmanov order is circular, we can proceed to a careful updating from edge to edge on the path $T_\ell^k(x_1x_k)$: when moving from the edge uv to the next edge vw, it is not necessary to compute again all the values A_i. This computation has to be done just for the A_i's such that $m(x_1,x_i,x_k) = v$ and $i \neq 1$, k; their number is the degree of v minus two. The total number of such operations related to the path $T_\ell^k(x_1x_k)$ is exactly k-2, for $k = 3,..., n$. Finally, though the

steps of Algorithm 5 seem more complicated than those of Algorithm 4, each step is at most $O(k)$ and this algorithm is $O(n^2)$ again.

As reported at the beginning of this section, such a time complexity is very good. Two $O(n^2)$ methods are proposed in Leclerc [17]. They are based on combinatorial properties of tree metrics, and not related with the least squares criterion.

Indeed, this time complexity $O(n^2)$ is not realistic (at least at the present state of this study): the initial choice of the elements x_1 and x_n is important in the use of algorithms 4 or 5, since it determines in fact a Yushmanov order among all the possible ones. Further studies or experimentations will be necessary to have an idea of the best strategy for this choice. In the experimental testings of the next subsection, we use the low complexity of Algorithms 4 and 5 to develop the alternative approach consisting of trying all the possible initial pairs (x_1,x_n). It gives two $O(n^4)$ methods, called Methods 1 and 2, based on Algorithms 4 and 5, respectively.

Both Methods 1 and 2 are completed with an adjustment of the lengths of the edges, once the topology of the obtained tree is fixed. Based on a least squares criterion on the differences between the obtained tree metric and the initial dissimilarity, this quadratic approximation is performed with the Gauss-Seidel method proposed in Barthélemy and Guénoche ([8], pp. 60-66; see for instance Ciarlet [8]). Since it requires an $O(n^4)$ time, it is mainly interesting for algorithms having at least this complexity. Even with this improvement, good results may hardly be expected from Method 1, because of the local character of Problem $P_{1,k}$. In Method 2, the approximation is done on the obtained trees that give the best result for the least squares criterion. The number of these trees is a small fixed one in Method M21 (in $O(n^4)$), and of order n in Method M22 (in $O(n^5)$).

4.3. Experimental results. Algorithms 1, 4 and 5 have been programmed in the C++ programming language and tested on a MS-DOS machine of IBM-PC type.

We use an evaluation method similar to that of Pruzanski, Tversky et Caroll [23], De Soete [12] and Gascuel et Levy [16]. Each data set is obtained as follows: first, an X-tree with n leaves and $2n$-3 arêtes is generated at random (for n = 12, 18, 24). The lengths of the edges are chosen at random from a uniform distribution on the real interval [0,1]. Then, the values of the corresponding tree metric are computed and normalized to have a unit variance, leading to a valued X-tree TT. A normal random noise of mean 0 and variance σ^2 = 0.1, 0.25, 0.5 is added to these values to obtain the distance d; in the rare cases where a negative value $d(x,y)$ results from these operations, this value is replaced with 0.01. A number of 100 data sets is generated for each pair of values of n and σ.

The results obtained from our methods M1 and M2 (with the variants M21 and M22 described above) are compared with those of the classical NJ method. We also consider the true tree TT which, contrary to the case of observed data, is known in these experimental ones.

The quality of the adjustement is evaluated by the means, computed on all the tests corresponding to each pair (n,σ), of two quantities:

1. The proportion of explained variance, as given by a formula of Pruzanski,Tversky and Caroll [23], where $m(d)$ is the mean value of d and t is the fitted tree metric:

$$\%\mathrm{Var} = \sqrt{\frac{\sum_{xy\in X^2}(d(xy)-t(xy))^2}{\sum_{xy\in X^2}(d(xy)-m(d))^2}}.$$

This quantity is also determined for the tree metric obtained after quadratic approximation (column "%Var+"). With this approximation, the NJ method becomes an $O(n^4)$ one.

2. The topological distance of Robinson and Foulds [24] between the true tree TT and the X-tree representation of t. It is a least move distance, the elementary move between two X-trees being the deletion or the addition of the split corresponding with an edge; that is, it is the symmetric difference metric on X-trees defined as sets of splits (Buneman [5]; see Barthélemy and Guénoche [2], ch. V). The distance between two trees is expressed as a percentage of the maximum value $3n$-6.

		$\sigma^2 = 0.1$			$\sigma^2 = 0.25$			$\sigma^2 = 0.5$		
$n = 12$	M1	88.14	93.54	12.76	73.93	84.98	18.06	61.49	75.84	26.57
	M21	92.62	93.66	10.47	83.24	85.44	16.00	73.86	76.96	21.63
	M22	92.62	93.69	9.80	83.24	85.56	14.70	73.86	77.17	20.90
	NJ	92.70	93.63	10.47	83.32	85.44	16.27	73.49	76.75	22.63
	TT	90.13	93.49		78.08	84.94		64.67	75.64	
$n = 18$	M1	84.80	92.26	17.58	68.65	82.58	27.12	51.52	69.89	37.48
	M21	91.33	92.57	12.19	80.89	83.55	20.77	68.30	72.12	31.75
	M22	91.33	92.61	11.64	80.89	83.73	19.73	68.30	72.55	29.08
	NJ	91.42	92.55	12.81	81.06	83.51	21.75	67.96	72.33	31.04
	TT	90.17	92.46		78.23	83.28		63.72	71.80	
$n = 24$	M1	82.74	91.28	23.64	65.46	80.05	37.83	46.47	66.24	46.92
	M21	90.42	91.83	16.62	79.04	82.07	28.13	64.62	69.65	37.02
	M22	90.42	91.93	13.54	79.04	82.34	28.08	64.62	70.24	35.60
	NJ	90.67	91.89	15.91	79.41	82.27	26.82	66.68	70.56	33.79
	TT	90.00	91.86		78.40	82.20		64.41	70.35	
		% VAR	%VAR+	RF	% VAR	%VAR+	RF	% VAR	%VAR+	RF

Table 2

The analysis of the results leads to the following observations: compared with the others, Method M1 is too elementary to give satisfactory results. The quadratic approximation is a very efficient tool for the improvement of the variance percentage. When the rank n is 12 or 18, Method M21 gives globally better results than NJ, and Method M22 is globally the best one in these tests. Nevertheless, method NJ is the

most robust when the size of the data and the variance of the noise both increase. For $n = 24$ and $\sigma^2 = 0.5$, it still gives the best results.

A further experiment is based on the data of Case [6] (immunological distances between nine species of frogs), frequently used for similar testing (see for instance Saitou and Nei [26] and Gascuel and Lévy [16]). Table 9 gives these distances.

	1	2	3	4	5	6	7	8
1: Aurora								
2: Boylii	10							
3: Cascadae	13	7						
4: Muscosa	12	7	7					
5: Temporaria	57	50	40	45				
6: Pretiosa	22	9	11	15	48			
7: Catesbiana	86	65	54	48	85	54		
8: Pipiens	89	67	66	49	83	55	54	
9: Tarahumarae	97	72	79	67	107	60	59	48

Table 3

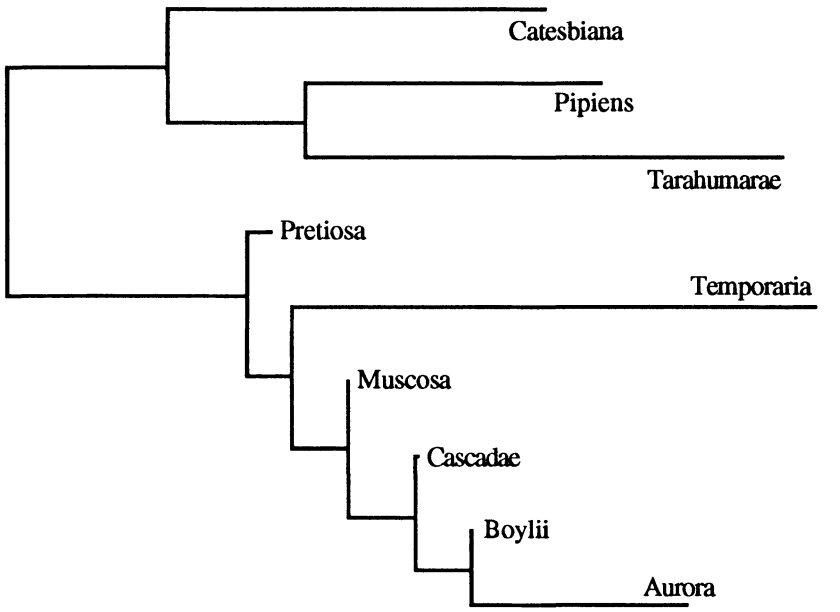


Figure 13

Five fitting methods are compared here: the methods M21 and M22 of this paper, the NJ method, and two other methods recognized to give generally good results: the method of scores (MS) of Luong [19] and Barthélemy and Guénoche [2], based on the grouping of pairs x, y such that $d(x,y) + d(z,w) < \max\{\ d(x,z) + d(y,w)\ ,\ d(x,w) + d(y,z)\ \}$ for a maximum number of pairs z, w (it may be considered as

a refinement of the ADDTREE method of Sattah and Tversky [27]); and the reduction method of Gascuel and Lévy [16], denoted here GL (it iteratively modifies the values of the dissimilarity towards a dissimilarity satisfying the four point-condition). These last two methods are $O(n^5)$. The best tree obtained by Method 2.2 and the corresponding tree metric are given in Figure 13 and Table 4.

Five criteria are used for the comparison of the methods. The combinatorial criterion NI (number of inversions) is the number of quadruples $xyzw$ such that a unique minimum among the three sums $d(x,y) + d(z,w)$, $d(x,z) + d(y,w)$ and $d(x,w) + d(y,z)$ disagrees with the configuration of the obtained X-tree. The criteria AAD , MAD and MSD respectively correspond to the average absolute difference, the maximum absolute difference and the mean squared difference between the values of the initial dissimilarity matrix and the obtained tree metric one. The criterion L , not available here for the reduction method, is the total length of the obtained valued X-tree, a short length being in agreement with the "parsimony principle" of phylogenetics. The criterion values given in Table 5 are those obtained after the quadratic approximation of the edge lengths mentioned in the previous subsection. In Method M21, the approximation is just done one time, on the best tree for the least squares criterion. Here, Method M22, where n trees are used for the approximation, provides a significant improvement to the few number of trees (here, three) considered in Method M21. The obtained tree is topologically an intermediate between those obtained by Saitou and Nei (also by method M21), on the one hand, and Gascuel and Lévy, on the other hand: it just differs from the former by the exchange of the places of *Muscosa* and *Cascadae*, and from the latter by the exchange of *Temporaria* and *Pretiosa*. Contrary to the Gascuel and Lévy method, the quadratic approximation is necessary to obtain a good fit.

1	2	3	4	5	6	7	8	
2	13.22							
3	16.61	3.39						
4	20.88	7.66	4.35					
5	60.66	47.43	44.13	39.78				
6	28.71	15.49	12.18	7.83	40.78			
7	76.03	62.81	59.50	55.15	88.12	50.39		
8	79.37	66.15	62.85	59.00	91.47	53.73	50.93	
9	90.52	77.30	73.99	69.64	102.61	64.87	62.07	48.00

Table 4

	ANI	AAD	MAD	MSD	L
Scores	n.a.	4.76	12.25	32.80	172.0
NJ	26	4.71	11.21	30.12	171.9
GL	23	4.52	10.05	28.95	n.a.
M21	26	4.71	11.21	30.12	171.9
M22	23	4.61	9.97	28.27	170.7

Table 5

5. Conclusion

The results and algorithms presented in this paper give evidence that Yushmanov orders are an interesting tool for the study of tree metrics. Among the questions arising about these orders, two of them seem especially interesting: the possible significance of Yushmanov orders for other types of dissimilarities; and their relations with the previously known combinatorial properties of tree metrics or X-trees (for instance the 4-ary relations characterized by Colonius and Schutze [9], the sets of splits of Buneman [5] and the relations with minimum spanning trees in Leclerc [17]).

Concerning the fitting algorithms of Section 4, the natural question is to device a way of preserving the low complexity of Algorithms 4 or 5. Here, this low complexity just allowed us to generate many good candidate trees, and to look for the best solutions among these candidates. Another direction of research is to generalize the method to other criteria, for instance the weighted least square one. An algorithm based on this criterion and sharing some features with those presented here, but without the use of circular orders, has been proposed by Makarenkov [20].

An important fact is the geometric signification, related to planar representations, of Yushmanov orders. Some uses of these orders for the drawing of trees, possibly directly from a dissimilarity array, may be expected: for instance, circular orders correspond to the so-called *hierarchical drawings* of an X-tree (Barthélemy and Guénoche [2], p.28). In such a drawing, inspired by the dendrograms of Numerical Taxonomy, the latent vertices are represented by horizontal lines, the upper one corresponding to the choice of a root. The edges of the tree are represented by vertical lines and no crossings are allowed. These orders, that can be obtained as right-left orders on the leaves in a hierarchical drawing, have been studied by Brossier [3] in the case of dendrograms; Figure 14 shows a hierarchical drawing of the tree of Figure 4, with the right-left order 10 9 4 3 6 5 8 7 2 1, which is circular. One may also expect that the algorithms described here could be modified in order to lead to a new family of low complexity methods of hierarchical classification, very different from the single linkage algorithm, which is, up to our knowledge, the only one in $O(n^2)$.

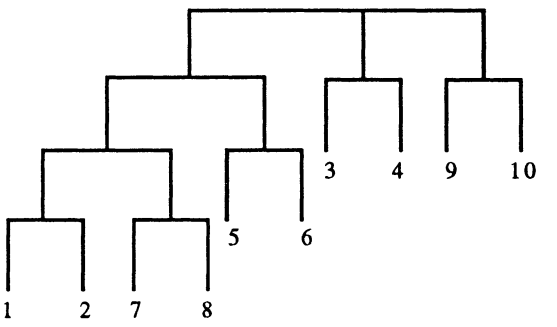


Figure 14

References

1. H.J. Bandelt, Recognition of tree metrics, *SIAM Journal on Discrete Mathematics* **3** (1990), 1-6.
2. J.P. Barthélemy, A. Guénoche, *Les arbres et les représentations des proximités*, Masson, Paris, 1988, transl. *Trees and proximity representations*,Wiley, New York, 1991.
3. G. Brossier, Représentation ordonnée des classifications hiérarchiques, *Statistique et Analyse des Données* **2** (1980), 31-44.
4. G. Brossier, Approximation des dissimilarités par des arbres additifs, *Mathématiques et Sciences humaines* **91** (1985), 5-21.
5. P. Buneman, The Recovery of Trees from Measures of Dissimilarity, in *Mathematics in Archaeological and Historical Sciences* (eds. F.R. Hodson, D.G. Kendall and P. Tautu), Edinburgh University Press, Edinburgh, 1971, 387-395.
6. S.M. Case, Biochemical systematics of members of the genus *Rana* native to wzstern North America, *Systematic Zoology* **27** (1978), 299-311.
7. S. Chaiken, A.K. Dewdney, P.J. Slater, An optimal diagonal tree-code, *SIAM Journal on Algebraic and Discrete Methods*, **4** (1983), 42-49.
8. P.G. Ciarlet, *Introduction à l'analyse numérique matricielle et à l'optimisation*, Masson, Paris, 1985.
9. H. Colonius, H.H. Schulze, Tree structure for proximity data, *British J. Math. Statist. Psychol.* **34** (1981), 167-180.
10. W.H.E. Day, Computational complexity of inferring phylogenies from dissimilarity matrices, *Bull. of mathematical Biology* **49** (1987), 461-467.
11. W.H.E. Day, Complexity theory: an introduction for practitioners of classification, in *Clustering and Classification* (eds P. Arabie, L.J. Hubert and G. De Soete), World Scientific Publ., River Edge, NJ, 1996, 199-233.
12. G. De Soete, A Least squares Algorithm for Fitting Additive Trees to Proximity Data, *Psychometrika* **48** (1983), 621-626.
13. G. De Soete, J.D. Caroll, Tree and other network models for representing proximity data, in *Clustering and Classification* (eds P. Arabie, L.J. Hubert and G. De Soete), World Scientific Publ., River Edge, NJ, 1996, 157-197.
14. A.K. Dewdney, Diagonal tree-codes, *Information and Control* **40** (1979), 234-239.
15. A.J. Dobson, Unrooted trees for numerical taxonomy, *J. Appl. Prob.* **11** (1974), 32-42.
16. O. Gascuel, D. Lévy, A reduction algorithm for approximating a (nonmetric) dissimilarity by a tree distance, *J. of Classification* **13** (1996), 129-155.
17. B. Leclerc, Minimum spanning trees for tree metrics: abridgements and adjustments, *J. of Classification* **12** (1995), 207-241.
18. B. Leclerc, V. Makarenkov, On some relations between 2-trees and tree metrics, Research Report n° 131, C.A.M.S., Paris, 1997, submitted.
19. X. Luong, Thesis, Université René Descartes, Paris, 1987.
20. V. Makarenkov, Deux algorithmes d'approximation d'une dissimilarité par une distance d'arbre au sens du critère des moindres carrés pondérés", *Les cahiers du C.A.M.S.*, n° 128, 1996.
21. M. Minoux, *Programmation mathématique, théorie et algorithmes*, Dunod, Paris, 1983.
22. A.N. Patrinos, S.L. Hakimi, The distance matrix of a graph and its tree realization, *Quart. Appl. Math.* **30** (1972), 255-269.

23. S. Pruzansky, A. Tversky, , J.D. Caroll, Spatial Versus Tree Representations of Proximity Data, *Psychometrika* **47** (1982), 3-19.

24. D.R. Robinson, L.R. Foulds, Comparison of phylogenetic trees, *Mathematical Biosciences* **53** (1981), 131-147.

25. M. Roux, Techniques of approximation for building two tree structures, in *Recent Devlopments in Clustering and Data Analysis* (eds. C. Hayashi, E. Diday, M. Jambu, N. Ohsumi), Academic Press, New York, 1988, 151-170.

26. N. Saitou, M. Nei, The neighbor-joining method: a new method for reconstructing phylogenetic trees, *Molecular Biology Evolution* **4** (1987), 406-425.

27. S. Sattah, A. Tversky, Additive similarity trees, *Psychometrika* **42** (1977), 319-345.

28. S.V. Yushmanov, Construction of a tree with p leaves from $2p$-3 elements of its distance matrix (russian), *Matematicheskie Zametki* **35** (1984), 877-887.

29. K. Zaretskii, Construction of a tree on the basis of a set of distances between its leaves (russian), *Uspekhi Mat. Nauk.* **20** (1965), 90-92.

CENTRE D'ANALYSE ET DE MATHÉMATIQUE SOCIALES, ÉCOLE DES HAUTES ÉTUDES EN SCIENCES SOCIALES, 54 Boulevard Raspail, F-75270 PARIS CEDEX 06, FRANCE.
E-mail Address: leclerc@ehess.fr

INSTITUTE OF CONTROL SCIENCES, 65 Profsoyuznaya, MOSCOW 117806, RUSSIA.

https://doi.org/10.1090/dimacs/037/12

DIMACS Series in Discrete Mathematics
and Theoretical Computer Science
Volume **37**, 1997

Estimation of Missing Distances in Path-Length Matrices: Problems and Solutions

Pierre-Alexandre Landry and François-Joseph Lapointe

ABSTRACT. Certain molecular techniques used in biological studies may lead to incomplete data sets, which greatly limit the number of tools available for phylogenetic analyses. However, lacunose (*i.e.* incomplete) distance matrices can often be estimated with relative accuracy from the available data. Several authors (De Soete, 1984a, 1984b; Lapointe and Kirsch, 1995; Landry et al., 1996) have explored the possibilities of estimating missing distances using two metric properties of path-length matrices: the ultrametric inequality (Hartigan, 1967) and the four-point condition (Buneman, 1971). In this paper, we have assessed the accuracy of the two estimation procedures using a simulation design based on noiseless additive matrices (the path-length matrices of trees obtained from DNA-hybridization data) to assess the strengths and weaknesses of each method. Results show that the additive procedure is the one to choose with low levels of missing cells, while the ultrametric one is more accurate with higher percentages of missing data. The simulations were repeated with ultrametric path-length matrices, where we show that the ultrametric procedure is more accurate in most cases. Given the complexity of the estimation problem, we make some suggestions as to how to get the most out of lacunose matrices in specific cases.

Introduction

It is not unusual in biological studies to obtain incomplete data sets, due to difficulties arising from the experimental methods, the lack of biological material, or to a combination of unpredictable factors. For example, phylogenetic analyses relying on DNA hybridization or comparative serology require n^2 comparisons among n taxa, which can limit the number of species included in these studies; it is not rare to end up with lacunose (*i.e.* incomplete) distance matrices in such cases. However, because distances are not independent from one another, metric properties can be used to estimate missing cells in distance matrices. Originally, De Soete (1984a) and Lapointe and Kirsch (1995) proposed using the ultrametric property (Hartigan, 1967) to fill lacunose matrices prior to phylogenetic reconstruction:

$$d(i, j) \leq \max[d(i, k); d(j, k)], \text{ for every } i, j, k. \tag{1}$$

1991 *Mathematics Subject Classification.* Primary 92D15, 92B10.

Key words and phrases. additive distances, DNA-hybridization data, lacunose matrices, missing data, path-length matrices, simulation study, topological recovery, tree metrics, ultrametric distances.

This work was supported by a NSERC scholarship to P.-A. Landry and by NSERC grant OGP0155251 to F.-J. Lapointe. The authors thank John A.W. Kirsch and an anonymous reviewer for their constructive comments on this manuscript.

Kirsch et al. (in press) have also shown that the ultrametric property can be applied to combine matrices representing overlapping sets of taxa. But biological data rarely fit the ultrametric model (*i.e.* not all species evolve at the same rate). A more general property of tree metrics, the four-point (or additive) condition (Buneman, 1971), could thus be used to estimate missing path-length distances more accurately:

$$(2) \qquad d(i, j) + d(k, l) \leq \max[d(i, k) + d(j, l); d(i, l) + d(j, k)], \text{ for every } i, j, k, l.$$

Landry et al. (1996) have observed that additive estimations of cells deleted from DNA-hybridization matrices were usually more accurate than the ultrametric estimations. Indeed, when the four-point condition is applied to estimate a distance deleted from complete additive or ultrametric matrices, its actual value can always be recovered provided that this missing distance is not between terminal sister-taxa (*i.e.* a terminal pair of leaves). This is also true for a distance missing from an ultrametric matrix and estimated with the corresponding ultrametric property. However, Landry et al. (1996) could not explain why the ultrametric property sometimes provided better estimates of missing distances in non-ultrametric lacunose matrices. The present paper addresses this specific question. The two different estimation methods will thus be applied and compared in a simulation design based on ultrametric and non-ultrametric additive matrices, in order to assess the relative efficiency of the competing procedures.

Methods

Simulation design. For comparison with previous studies, we have selected the same DNA-hybridization matrices [$n = 7$: Dasyuridae (Marsupialia); Kirsch et al. 1990, $n = 9$: hummingbirds (Apodiformes); Bleiweiss et al. 1994, $n = 11$: Didelphidae (Marsupialia); Kirsch et al. 1995a, $n = 13$: Diprotodontia (Marsupialia); Springer and Kirsch 1991, $n = 15$ Pteropodidae (Chiroptera); Kirsch et al. 1995b] analyzed by Lapointe and Kirsch (1995) and Landry et al. (1996). In the present case, we were interested in the phylogenies reconstructed from the complete matrices. An additive tree was thus derived from each DNA-hybridization matrix using the FITCH program implemented in the PHYLIP package, version 3.5c (Felsenstein, 1993). Ultrametric trees were also obtained from the same matrices with the KITSCH program (Felsenstein, 1993). FITCH and KITSCH both are least-squares algorithms to compute the optimal additive or ultrametric tree respectively from a given distance matrix. The corresponding path-length matrices computed from those trees were then used in the simulations. The experimental design follows Landry et al. (1996):

(a) a lacunose matrix is created by randomly removing known values from the complete matrix;
(b) missing cells are estimated using either the ultrametric or the four-point condition;
(c) a phylogeny is reconstructed from the filled matrix, using either FITCH or KITSCH;
(d) the corresponding path-length matrix is computed from the phylogeny;
(e) each tree is compared with the corresponding phylogeny derived from the complete matrix.

The lacunose matrices were generated with an increasing percentage of missing cells [$P = \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6\}$], deleting reciprocal distances [*i.e.* $d(i, j)$ and $d(j, i)$] from the complete square matrix in each case. One hundred replicates were generated for each combination of n and P. In a first series of simulations, matrices were filled using the ultrametric property (eq. 1). Then, we repeated the simulations with the same set of

matrices (removing exactly the same cells from the corresponding matrices), this time using the four-point condition (eq. 2) to fill the empty cells. Deletions of the same distances in both series was deemed necessary to control for sampling errors due to the randomization. The simulations were repeated twice: first with non-ultrametric additive matrices, and then with ultrametric ones.

Estimation procedure. In order to estimate missing cells in the lacunose distance matrices, we used the program RECALL (available from the authors upon request) based on the following algorithm: i) the lacunose matrix is read, and missing distances are identified; ii) for each missing cell encountered, every possible quartet (additive procedure) or triplet (ultrametric procedure) of relevant taxa is used to estimate the missing distance; iii) the minimum of all the estimates computed is returned as the final estimate of the missing cell; iv) the process is repeated iteratively until the matrix is completely filled.

The ultrametric procedure will usually fill a matrix in a single pass through the data, even when numerous cells are missing. On the other hand, the additive procedure will not always work with very sparse distance matrices. Indeed, when no quartet contains a sufficient amount of information (*i.e.* five distances out of six), no estimate will be returned. In such cases, the ultrametric property needs to be called to fill the minimum number of cells required to apply the four-point condition. To do so, one of the missing cell is selected at random and is estimated with the ultrametric procedure. An iterative procedure is thus implemented by calling the ultrametric and additive properties recursively until the matrix is filled.

Recovery indices. Matrices obtained by the different procedures were compared for metric and topological recovery. First, we computed correlations between path-length matrices associated with the trees derived from the original and estimated data; this was done to assess whether the estimation procedure could recover the actual path-length distances. Then, we compared the topology of the original tree with those derived from the estimated matrices using the partition metric (Robinson and Foulds, 1981; computations made in PAUP: Swofford, 1993). In order to compare the topological recovery for matrices of different sizes, the partition metric (p_m) was standardized by its maximal possible value ($2n - 6$) for each matrix (Steel and Penny, 1993). The complement of this standardized statistic (*i.e.* $1 - p_m$) was then taken as an index of topological similarity, with a maximal value of one for topologically identical trees, and a value of zero for the most different possible pairs of trees. The average metric recovery values and the median of the topological recovery values were finally recorded for comparison purposes.

Results

Additive matrices. The results of the simulations on additive matrices are presented in figure 1. As already stated by Lapointe and Kirsch (1995) and Landry et al. (1996), it is again clear from our results that there is no relationship between recovery values and matrix size. Therefore, we have to assume that the differences among matrices are due to intrinsic factors such as their relative branch lengths or topologies of the phylogenies compared. For example, the topology of a tree with short internal branches is known to be more difficult to recover then one with long internodes (see Lapointe and Kirsch, 1995).

Metric recovery. Our most striking result (see fig. 1A) is the superiority of the additive procedure in recovering path-length distances for most matrices ($n = 7$, 9, and 15). In those cases, the recovery levels for additive estimates are always higher than, or

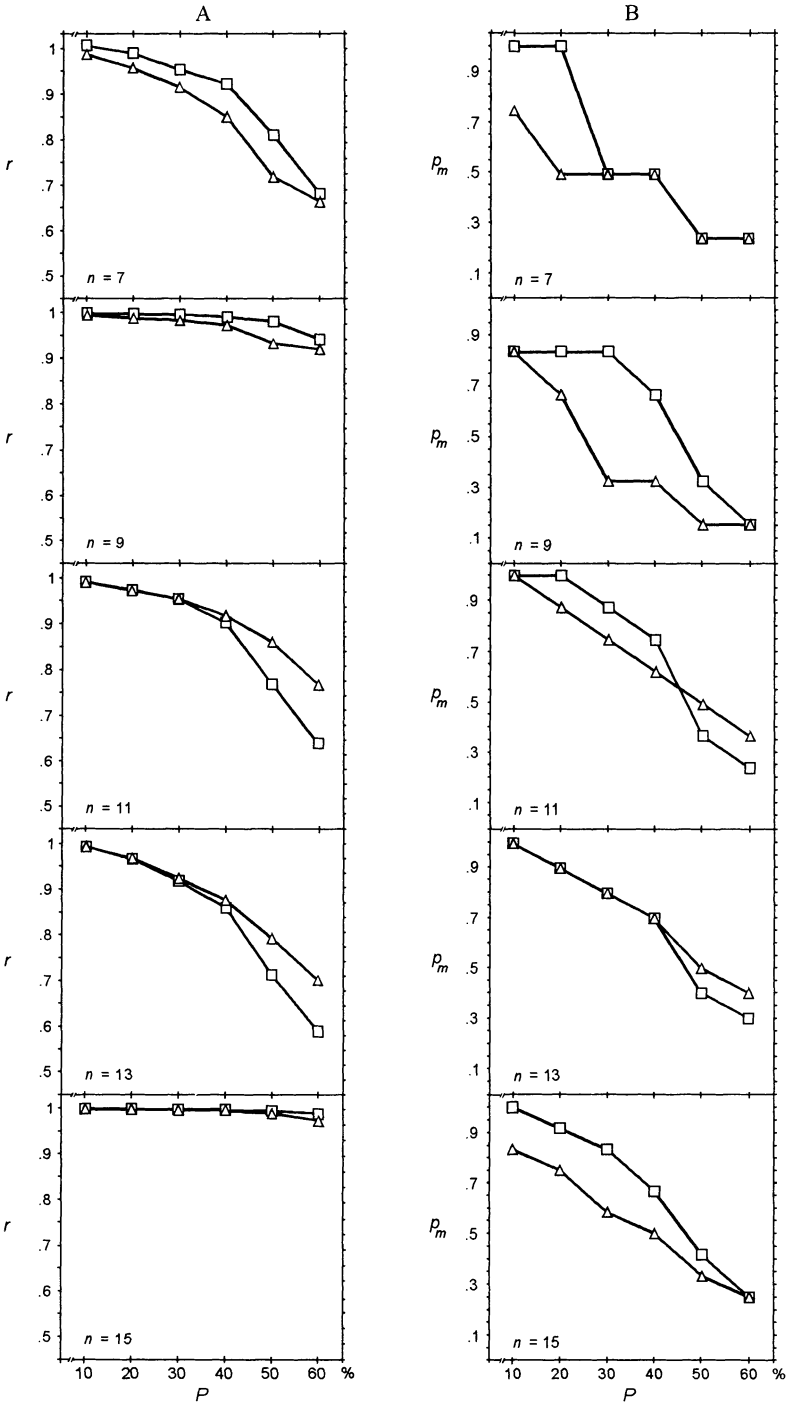


FIGURE 1.—Metric and topological recovery values obtained for non-ultrametric additive matrices of increasing size (n = 7, 9, 11, 13, 15), for varying percentages of estimated cells (P = 0.1, 0.2, 0.3, 0.4, 0.5, 0.6), using the ultrametric inequality [open triangles] and the four-point condition [open squares]. A, path-length correlations (r); B, standardized partition metric values (p_m).

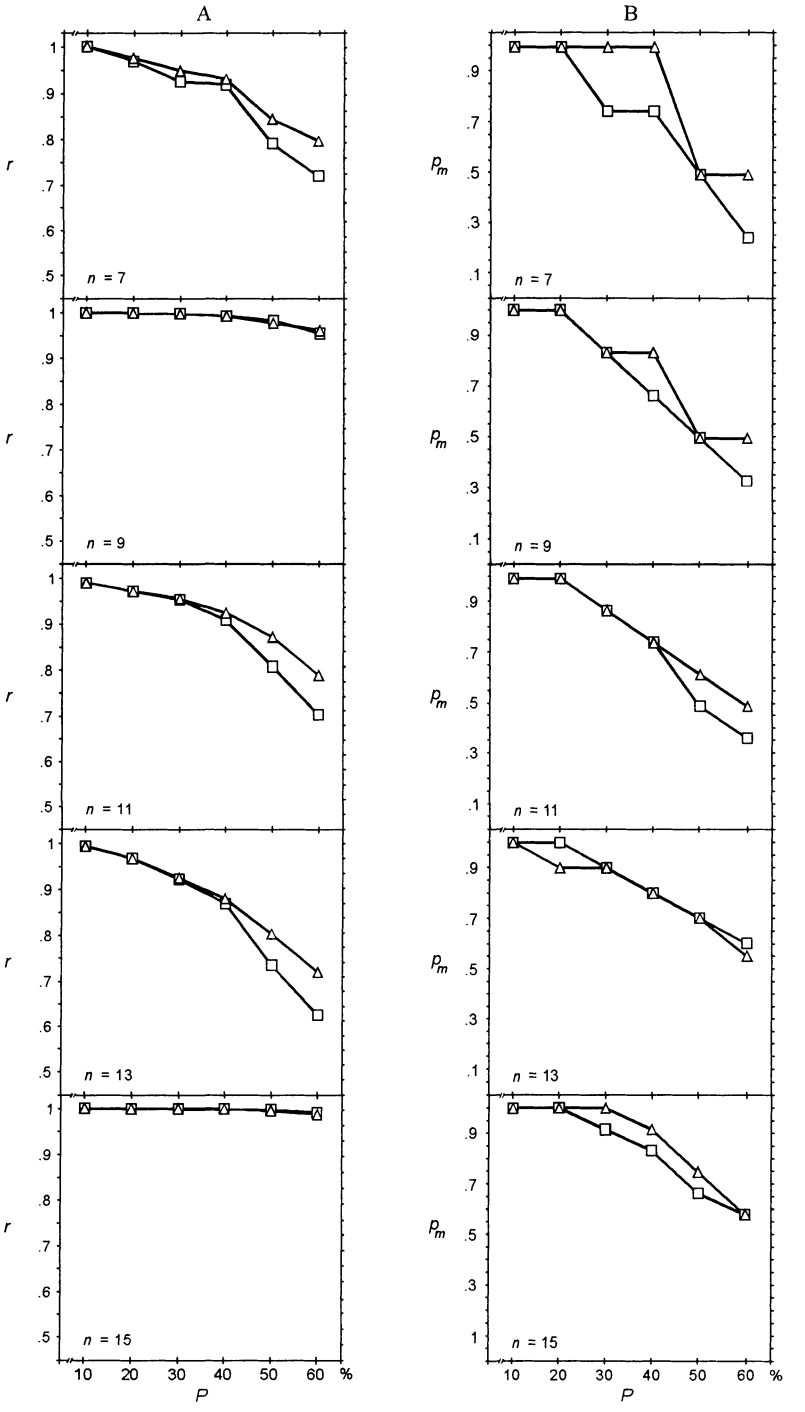


FIGURE 2.—Metric and topological recovery values obtained for ultrametric matrices of increasing size (n = 7, 9, 11, 13, 15), for varying percentages of estimated cells (P = 0.1, 0.2, 0.3, 0.4, 0.5, 0.6), using the ultrametric inequality [open triangles] and the four-point condition [open squares]. A, path-length correlations (r); B, standardized partition metric values (p_m).

equal to, those based on ultrametric estimations. These results were expected, since the four-point condition is less restrictive regarding rate differences than the ultrametric inequality, although additive estimation does require more information. Interestingly, one notices that the correlation values for additive and ultrametric estimations converge for the smaller matrices (n = 7, 9), as the percentage of missing cells increases; this is not true for some of the larger matrices (n = 11, 13,). This result, however, can be partly explained by the estimation algorithm. When the number of missing cells increases, we know that it becomes more and more difficult to compute an additive estimate. The program must sometimes rely on the ultrametric inequality to fill the minimum number of cells needed to subsequently call the four-point condition. This situation is more likely to happen in smaller matrices, because the number of possible quartets is function of the size of the matrix. Therefore, the larger the matrix, the fewer the ultrametric estimations (see Landry et al., 1996). This circumstance can explain why the convergence of the two correlation curves is not observed for the largest matrix (n = 15); in this particular case, the ultrametric procedure was never called when less than 50% of the cells needed to be estimated.

It is odd that both estimation methods return equivalent recovery values for two matrices (n = 11, 13). Even more puzzling is the fact that the ultrametric estimates are better when more than 30% of the cells are missing. With the same matrices, Lapointe and Kirsch (1995) have shown that accuracy of the ultrametric estimations depends on the ultrametricity of a given matrix. Indeed, a particular additive tree may be nearly ultrametric. If a path-length matrix is close to being ultrametric, it is conceivable that one might obtain accurate estimates when the ultrametric property is called for (see the **Ultrametric matrices** section below). We thus compared the five additive matrices (FITCH trees) with the competing ultrametric ones (KITSCH trees) to measure the fit between the corresponding trees. Ultrametricity of a given additive matrix was measured as the standardized sum of squared differences [sSS = 100*SS/n (n - 1)] of the distances between the scaled path-length matrices corresponding to FITCH and KITSCH trees. This measure clearly illustrates that the more ultrametric matrices (n = 11, sSS = 0.129; n = 13, sSS = 0.099), are exactly those for which the ultrametric procedure is more accurate for higher values of P, while the other matrices (n = 7, sSS = 7.969; n = 9, sSS = 3.362; n = 15, sSS = 1.492) are more accurately filled with the four-point condition (see fig. 3).

According to the tree metric properties, the use of the four-point condition should perfectly recover ultrametric distances with noiseless data. However, this is only true when the missing distance is not between terminal sister-taxa. In fact, it is very likely that the differences observed between the two estimation procedures reflect the errors made when a distance between such sister-taxa is estimated. For an ultrametric matrix, the tree derived from the estimated matrix will bear a trichotomy involving the sister-taxa (Lapointe and Kirsch, 1995). In the case of non-ultrametric matrices, the estimation will vary greatly, depending on the available set of quartets. We observed that the additive procedure is usually more precise than its ultrametric counterpart, but the errors made using the four-point condition are often more important. Because it is more conservative, the ultrametric procedure will also make smaller errors, on average, than the additive procedure. Those errors are cumulative as missing distances are estimated from previously filled cells. Larger errors, such as those produced by the four-point condition, can then have a larger impact, especially for very sparse matrices.

Topological recovery. For topological recovery (fig. 1B), we also observe the absence of relationship between the size of the matrix and the recovery levels. However, trees derived from matrices which were filled with the additive procedure revealed fewer topological errors. Also, there seems to be no clear relationship between metric and topological recovery. Yet, one notices that the procedure (additive or ultrametric) with the best metric recovery levels is generally the one with the best topological recovery levels. Interestingly, even when metric recovery values are quite similar for both estimation procedures ($n = 9$, 11 and 15), topological recovery is usually better for the additive one. These results also indicate that slight variations in the estimated distances may lead to topological discrepancies otherwise trivial in terms of metric recovery, which is surely related to the short internodes in the 15-taxon tree (see fig. 3B).

Ultrametric matrices. Repeating the simulations with ultrametric matrices assesses whether the additive procedure could do as well in those cases. These cross-simulations could help us understand the relative behavior of the different estimation procedures under various conditions; *i.e.* lacunose ultrametric matrices estimated with ultrametric or additive criteria, as opposed to lacunose non-ultrametric additive matrices estimated with the same procedures. The results for both metric and topological recovery are presented in figure 2.

Metric recovery. Our results clearly indicate that the estimations based on the ultrametric method are generally more accurate for ultrametric matrices than those obtained with the additive procedure (fig. 2A). This represents another indication that the ultrametric procedure may be better able to estimate distances between terminal sister-taxa. Also, one notices again that matrix size is not related to average recovery levels. The matrices which show higher recovery values in the first series of simulations (see fig. 1A) once more exhibit higher correlation values. However, the recovery levels obtained with both procedures are more similar than were those observed with the additive matrices, mainly because ultrametric matrices represent particular types of additive matrices. Indeed, Landry et al. (1996) have shown that any missing ultrametric distance not involving sister-taxa can be perfectly recovered with both procedures.

Topological recovery. Topological recovery results convey the same information as in the case of additive matrices, *i.e.* that there is no relationship between matrix size and topological recovery (fig. 2B). Also, metric and topological recovery levels are not clearly related, as in the previous simulations based on additive matrices. As expected, the ultrametric procedure outperforms the additive method in all cases but one ($n = 13$). Nevertheless, the differences between the two procedures are smaller than those obtained with additive matrices (see fig. 1B). Here again, this should be obvious because ultrametric distances can be estimated using the four-point condition, the reverse not being true.

Discussion

The problem of filling lacunose matrices rapidly becomes very difficult when many cells are missing. Because distances are not independent from one another, attempting to state clearly the factors affecting the results of the estimations can be quite hazardous. For example, when a missing cell is filled, it is likely to be estimated incorrectly if terminal sister-taxa are involved. But how to know *a priori* if a distance between a terminal pair of taxa is missing? Also, if trees bearing short branches are considered, one is very likely to obtain incorrect topologies from estimated matrices, despite accurate metric recovery. How to assess topological recovery in such superficially paradoxical cases? Another

concern is the fact that the four-point condition does not perform as well when the matrix is partly ultrametric. Would it be possible to use different procedures to estimate different parts of the matrix? When more than one cell is missing, all of these problems become more important as previously estimated cells are used for subsequent estimations. Therefore, one can wonder if the order in which missing cells are estimated will have an effect on the overall recovery levels. These are some of the questions one faces when missing cells are estimated. The following sections present solutions to these problems, or ways to identify them.

The terminal sister-taxa problem. Terminal sister-taxa represent the worst problem when estimating missing distances; there is no exact way to recover perfectly such missing distances. However, while accepting that the estimation of a distance between a terminal pair is likely to be inaccurate, it can be wiser to estimate those cells last to minimize the errors in the subsequent estimations of other missing distances. But, in order to do so, one would need to identify sister-taxa without knowing the topology of the tree *a priori*. How to solve that problem? One suggestion is to look at the phylogeny derived from the filled matrix to identify terminal pairs, and check whether those involve taxa among which distances were estimated. If distant taxa are involved, or if an intermediate taxon is lying on the path between a pair fo which a distance was estimated (Lapointe and Kirsch, 1995), that estimate is probably accurate. On the other hand, if the distance estimated is between closely-related taxa, it is always possible that a terminal sister-pair was incorrectly recovered (especially if the taxa involved belong to the same clade).

The short internodes problem. As stated earlier, topological recovery of the estimated matrices depends on the metricity of the corresponding tree. Trees that present short internodes are known to be less stable, because they are more vulnerable to small estimation errors which may lead to topological inversions, especially with experimental data (Lapointe and Kirsch, 1995). Topological recovery can be poor, even though metric recovery is good; this paradox can be explained by the relatively short internodes in the corresponding tree (fig. 3B). On the other hand, even low levels of metric recovery sometimes correspond to high topological recovery levels, as in the case of the $n = 11$ matrix; that particular phylogeny contains longer internal branches which confer more stability, even though metric recovery declines quickly with P. On the other hand, the $n = 15$ matrix shows almost perfect metric recovery over all simulations, but amongst the poorest topological recovery. The problem is to predict the topological accuracy of an estimated matrix. Assuming that metric recovery is not related to absolute branch lengths, we expect misplaced taxa (or entire clades) still to be supported by short internodes, as if they were positioned correctly. It is therefore recommended to check whether the estimated distances are among clades supported by short internodes. If so, one should treat the derived phylogeny with caution.

The additive decomposition problem. Any additive matrix can be decomposed into an ultrametric and a star component (Farris et al., 1970). This is also true for lacunose additive matrices, provided that at least one row (or column) is complete. Following decomposition, one can estimate the missing cells in the ultrametric component using the ultrametric inequality, before adding the star component to recover the additive matrix. We have observed that this procedure returns exactly the same distances as those estimated with the four-point condition. Nevertheless, this property could be interesting when combining matrices, especially if one wishes to suture an ultrametric matrix with an

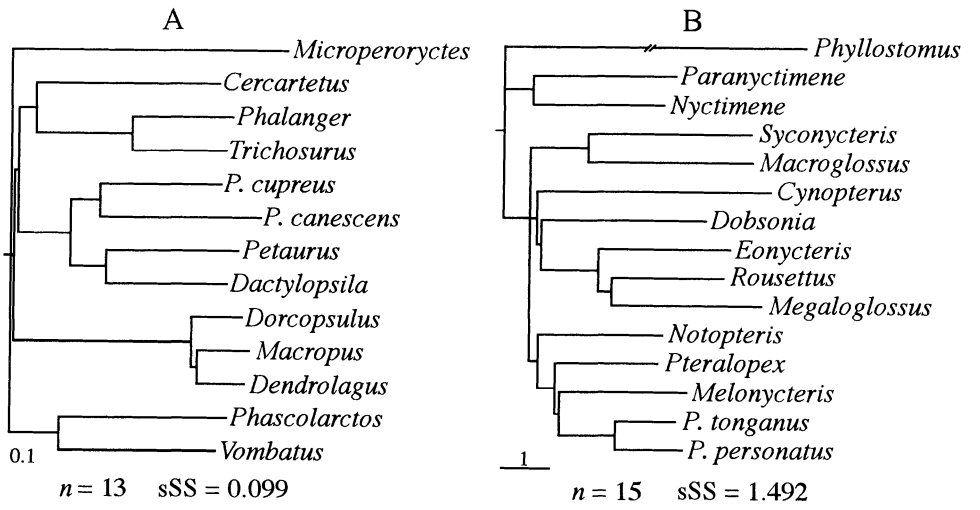


FIGURE 3.— Examples of phylogenies derived from the complete DNA-hybridization matrices [A: n = 13: Diprotodontia (Marsupialia); Springer and Kirsch 1991, B: n = 15 Pteropodidae (Chiroptera); Kirsch et al. 1995b]. The standardized sum of squares (sSS) reflects the ultrametricity of the corresponding trees. A shows a nearly-ultrametric additive tree, while B is not close to being ultrametric. Also note the very short internodes in B (all computations made in FITCH of the PHYLIP package; Felsenstein, 1993).

additive one. For instance, one could extract the ultrametric component from the additive matrix to combine it with an ultrametric one using the ultrametric property, and then add the star component to the decomposed matrix at the end. This approach could simplify the computations because triplets instead of quartets would have to be considered. Decomposition can be difficult, however, with very sparse matrices lacking complete rows of distances.

The filling order problem. Filling of the same lacunose matrices with permuted labels has confirmed that the order in which the estimations are done may induce minor variations in metric recovery; this can in turn lead to major differences at the topological level. These topological discrepancies should be even more likely with noisy data (e.g. DNA-hybridization distances) or when terminal sister-taxa are involved (see above). In such a case, one should be able to impose a filling order to obtain accurate estimates. It would be interesting to develop an algorithm that computes accurate estimates regardless of the filling order. Yet, one should notice that it is always possible to randomize the labeling order to alleviate the bias induced by the positions of the missing distances in the matrix. However, this does not guarantee that the estimates obtained will be more accurate. One should rely on an *ad hoc* criterion to select the best estimates in such cases. Interestingly, and to emphasize this point once more, matrices with higher metric recovery levels are not necessarily the ones with the fewer topological errors. It would be of great interest to repeat the estimation process a number of times to generate several trees; a consensus of those trees could then be computed to synthesize the results. This should lead to a more accurate phylogeny, on average, than any tree based on a particular estimation-order of the lacunose matrix.

Conclusion

This paper evaluates the recovery levels of missing distances estimated in path-length matrices using the ultrametric and four-point conditions. Our study also extends previous comparisons performed on DNA-hybridization data to path-length matrices corresponding to phylogenetic trees. The results of the simulations indicate that it is usually more accurate to estimate missing cells using the additive procedure, when less than 30% of the cells are missing. However, the accuracy of the additive procedure is related to the ultrametricity of the matrix estimated; that implies that nearly ultrametric trees will be better recovered using the ultrametric inequality, for values of P greater than 0.30. One should still bear in mind that the simulations were intentionally extended to high percentages of missing cells to compare the two procedures under extreme conditions. With real data, it is quite rare to encounter large numbers of missing cells, and users should be aware that the estimations become less reliable as P increases. On the basis of the present and previous results with experimental data (see Landry et al., 1996), we recommend giving preference to the additive procedure in most cases. Finally, let us reiterate that the recovery levels are data-dependent and are affected by intrinsic properties of the matrices considered. In any case, missing data are to be treated with caution and trees derived from filled matrices, such as the implied internodal lengths, should always be validated thoroughly (see Kirsch et al., in press).

References

1. Bleiwess, R., J. A. W. Kirsch, and J. C. Matheus. *DNA hybridization evidence for subfamily structure among hummingbirds.*, Auk 111 (1994), 8-19.
2. Buneman, P. *The recovery of trees from measures of dissimilarity*. Pp. 387-395 in F. R. Hudson D. G. Kendall, and P. Tautu, eds. Mathematics in archeological and historical sciences. Edinburgh University Press, Edinburgh, 1971.
3. De Soete, G. *Utrametric tree representations of incomplete dissimilarity data,* J. Classif. 1 (1984a), 235-242.
4. —. *Additive-tree representations of incomplete dissimilarity data*, Qual. Quantity 18 (1984b), 387-393.
5. Felsenstein, J. *PHYLIP: Phylogeny Inference Package*, version 3.5c. Distributed by the author. University of Washington, Seattle, Washington, 1993.
6. Farris, J. S., A. G. Kluge, and M. J. Eckart, *A numerical approach to phylogenetic systematics*, Syst. Zool. 19 (1970), 172-189
7. Hartigan, J. A. *Representation of similarity matrices by tree,* J. Am. Stat. Assoc. 62 (1967), 1140-1158.
8. Kirsch, J. A. W., A. W. Dickerman, and O. A. Reig. *DNA/DNA hybridization studies of carnivorous marsupials. IV. Intergeneric relationships of the opossums (Didelphidae)*, Marmosiana 1 (1995a), 57-78
9. Kirsch, J. A. W., T. F. Flannery, M. S. Springer, and F.-J. Lapointe. *Phylogeny of the Pteropodidae (Mammalia: Chiroptera) based on DNA hybridization, with evidence for bat monophyly,* Aust. J. Zool. 43 (1995b), 557-582.
10. Kirsch, J. A. W., C. Krajewski, M. S. Springer, and M. Archer. *DNA/DNA hybridisation studies of carnivorous marsupials. II. Relationships among dasyurids (Marsupialia),* Aust. J. Zool. 38 (1990), 673-696.
11. Landry, P.-A., F.-J. Lapointe, and J. A. W. Kirsch. *Estimating phylogenies from lacunose distance matrices: Additive is superior to ultrametric estimation*, Mol. Biol. Evol. 13 (1996), 818-823.
12. Lapointe, F.-J., and J. A. W. Kirsch. *Estimating phylogenies from lacunose distance matrices, with special reference to DNA hybridization data*, Mol. Biol. Evol. 12 (1995), 266-284.
13. Robinson, D. F., and L. R. Foulds. *Comparison of phylogenetic trees,* Math. Biosci. 53 (1981), 131-147.
14. Springer, M. S., and J. A. W. Kirsch. *DNA hybridization, the compression effect, and the radiation of diprotodontian marsupials,* Syst. Zool. 40 (1991), 131-151.
15. Steel, M. A., and D. Penny. *Distributions of tree comparison metrics – some new results*, Syst. Biol. 42 (1993), 126-141.
16. Swofford, D. L. *PAUP: Phylogenetic analysis using parsimony*, version 3.1.1. Distributed by the Illinois Natural History Survey, Champaign, Illinois, 1993.

DÉPARTEMENT DE SCIENCES BIOLOGIQUES, UNIVERSITÉ DE MONTRÉAL, MONTRÉAL, CANADA.
PHONE: (514) 343-7999, FAX: (514) 343-2293.
E-mail addresses: landryp@ere.umontreal.ca, lapoinf@ere.umontreal.ca.

https://doi.org/10.1090/dimacs/037/13

DIMACS Series in Discrete Mathematics
and Theoretical Computer Science
Volume **37**, 1997

Complexity Issues in Hierarchical Optimization

P.M. Pardalos and X. Deng

ABSTRACT. The study of hierarchy occurring in biological structures reveals interesting properties as well as limitations due to stability properties of molecules. Hierarchical optimization can be used to study properties of hierarchical designs that occur in biological structures. In recent years hierarchical decision processes received an increasing interest in optimization within the area of multi-level programming. Although many problems of practical interest can be formulated as multi-level programming problems, most of these problems are NP-hard from the complexity point of view. We are going to discuss recent results regarding the complexity of various types of multi-level problems and the importance of these results in future algorithmic developments.

1. Introduction

Hierarchical designs are found in many complex systems and, in particular, in biological systems. Nature makes very different biological systems (that have specific hierarchical composite structures) out of very similar molecular constituents. In hierarchical architectural designs of biological systems, organization is controlled on length scales ranging from the molecular to macroscopic. These hierarchical architectures rely on critical interfaces that link structural elements measured on disparate scale [**2, 14, 22, 36, 33**]. First, the structures are organized in discrete levels. Second, the levels of structural organization are held together by specific interactions between components. And finally, these interacting levels are organized into an oriented distinct hierarchical composite system of specific function.

The study of hierarchy occurring in biological structures reveals interesting properties as well as limitations due to different properties of molecules.

Hierarchical optimization can be used to study properties of the hierarchical designs. In hierarchical optimization, the constraint domain is implicitly determined by a series of optimization problems which must be solved in a predetermined sequence. In recent years many attempts have been made to develop algorithms for solving hierarchical optimization problems [**1, 25, 23**].

The main focus of this paper is to provide an overview of several complexity issues related to hierarchical optimization. By using complexity we can analyze the intrinsic difficulty of optimization problems and reveal surprising connections

---

1991 *Mathematics Subject Classification.* Primary 90B80, 90C20, 90C35, 90C27; Secondary 65H20, 65K05.

This research was partially funded by DIMACS and National Science Foundation grant BIR-9505919.

among many other optimization problems and their solutions [**2**]. Moreover, different complexity results for different structures may provide us with new insight into the mechanisms of these structures [**29**]. In the study of game theoretical foundation of economics, for example, computational complexity has been suggested as one important factor of "bounded rationality" [**26, 28, 9**], a concept introduced by Simon [**32**] (and may even be traced back to Clark [**7**] as suggested by Radner [**31**]).

2. Hierarchical optimization

Hierarchical (or multi-level) optimization was first defined and studied in [**5, 6**] as a generalization of mathematical programming. The simplest two-level (or bilevel) programming problem describes a hierarchical system which is composed of two levels of decision makers and is stated as follows:

$$\textbf{(BP)} \quad \min_{y \in Y} \quad \varphi(x(y), y) \tag{1}$$

$$\text{subject to} \quad \psi(x(y), y) \leq 0 \tag{2}$$

$$\text{where} \quad x(y) \quad = \quad \arg\min_{x \in X} f(x, y) \tag{3}$$

$$\text{subject to} \quad g(x, y) \leq 0, \tag{4}$$

where $X \subset R^n$ and $Y \subset R^m$ are closed sets, $\psi : X \times Y \to R^p$ and $g : X \times Y \to R^q$ are multifunctions, φ and f are real-valued functions. The set $\mathcal{S} = \{(x, y) : x \in X, y \in Y, \psi(x, y) \leq 0, g(x, y) \leq 0\}$ is the *constraint set* of **BP** (the vectors 0 have the appropriate dimensions). For fixed $y \in Y$, define the set $X(y) = \{x \in X : g(x, y) \leq 0\}$ and let $\mathcal{R}(y) = \{x \in X : x \in \arg\min_{w \in X(y)} f(w, y)\}$. The *feasible set* of **BP** is $\mathcal{F} = \{(x, y) \in \mathcal{S} : x \in \mathcal{R}(y)\}$. A feasible point $(x^\star, y^\star) \in \mathcal{F}$ is often called a Stackelberg equilibrium if $\varphi(x^\star, y^\star) \leq \varphi(x, y)$ for all $(x, y) \in \mathcal{F}$.

Multi-level programming problems have been studied extensively in their general setting during the last decade. For a continuously updated bibliography on the subject see [**35**]. In general, hierarchical optimization problems are nonconvex and therefore, it is not easy to find globally optimal solutions [**16, 17**]. Moreover, suboptimal solutions may lead to both theoretical and real-world paradoxes (as for instance in the case of network design problems [**3**]).

Many algorithmic developments are based on the properties of special cases of **BP** (and the more general problem) and reformulations to equivalent or approximating models, presumably more tractable. Most of the exact methods are based on branch and bound or cutting plane techniques and can handle only moderate size problems.

3. Complexity issues

3.1. Complexity of finding a global solution. Hierarchical optimization problems are inherently very difficult and complex. This fact has been confirmed by numerous studies of what might be considered the simplest version of **BP**, that is, the *linear* bilevel optimization problem, i.e. the functions in (1)-(2) and in (3)-(4) are linear. In 1985, Jeroslow [**18**] has shown that the linear **BP** is *NP-hard.* His results were subsequently confirmed and simplified in [**4**] and strengthened in [**15**] where it was demonstrated that the linear **BP** is *strongly NP-hard.*

On the other hand, these results only state the problem is difficult. There may not be any limitation on how hard this problem can be in the general case. Deng

and Papadimitriou [**10**] have recently shown that the problem cannot be harder: it is $F\Delta_2^P$-complete. In other words, given an NP-oracle, the problem can be solved in polynomial time with a polynomial number of calls to the oracle [**13, 27**]. The same result holds even when there are several sublinear programming problems at the lower level with shared constraints as long as the lower level agents reach a Nash Equilibrium point best for the upper level linear program (the focal point) [**10**].

Dubas et al. have considered the complexity issues of different possible responses, including both the worst case response and the best case response to the upper level agent (the leader), of the lower level agent (the follower), and among multiple choices of the same optimal value for itself [**12**].

In general, proof techniques in NP-hardness of BP can be summarized as follows. Consider an NP-hard problem for the reduction. We first write it as a feasibility problem of an integer program of $0 - -1$ values for the variables, say $X = \{x_1, x_2, \cdots, x_n\}$. To force these variables to be of integer value in a bi-level linear program, one introduces a set of constraints and objective functions for the upper level and the lower level so that the upper level would have to choose integer values for the linear integer program to maximize its objective function, taking into consideration how the lower level program will react to its decisions. For example, we can introduce another set of variables $Z = \{z_1, z_2, \cdots, z_n\}$ and the set of constraints $z_i \leq x_i, z_i \leq 1 - x_i$. The objective function of the upper level is chosen to be $\min \sum_{i=1}^n z_i$. We choose the objective function of the lower level to be $\max \sum_{i=1}^n z_i$. Make the control variables of the upper level to be X, and the control variables of of the lower level to be Z. The lower level will thus choose $z_i = \max\{x_i, 1 - x_i\}$. Because of this, the upper level will choose $x_i \in \{0, 1\}$ in order to minimize $\sum_{i=1}^n z_i$.

Generally speaking, **BP** belongs to the realm of global optimization problems and this has been demonstrated through various transformations by a number of researchers, see e.g. [**23, 25, 24**].

In [**15**] it has been shown that the linear **BP** is *strongly NP-hard*, by considering the complexity of a special case of a linear max-min optimization problem. Max-min linear programs can be formulated as convex maximization (or concave minimization problems). The computational complexity of min-max optimization problems has been extensively studied in [**19**]. These problems are naturally characterized by Π_2^P, the second level of polynomial-time hierarchy.

3.2. Complexity of local search. Computing locally optimal solutions is presumably easier than finding globally optimal solutions. However, from the complexity point of view it has been shown (Pardalos and Schnitger [**30**]) that the problem of checking local optimality for a feasible point and the problem of checking whether a local minimum is strict, are NP-hard. These complexity results remain true even for instances of quadratic programming with a simple structure in the constraints and the objective. These results have been proven by a reduction from the classical 3-Satisfiability problem [**13**]. This reduction has been reformulated to show that checking local optimality in bilevel programming is NP-hard by Vicente et al. [**34**].

This modification is similar to the one discussed before. The solution of the quadratic program is of some discrete set of values and the value of the objective function is zero, if and only if the 3SAT instance is satisfiable. To transfer it into

a bilevel program, each quadratic term, (say xy) of the objective function is split into two constraints (say $z \leq x, z \leq y$). The upper level will minimize $\sum_{i=1}^{n} z_i$ and the lower level will maximize it. The upper level controls the original variables of the quadratic program, the lower level controls the new variables $z_1, z_2, \cdots, z_n$.

Several issues remain open. Is it possible that some locally optimal solutions are easy to find? Could we find large classes of problems for which locally optimal solutions can be found in polynomial time?

3.3. Approximation. Jeroslow has observed that for any constant factor $c > 0$, it remains NP-hard to find a solution within a multiplicative factor of c times the optimum. Applying the tools of MAX SNP completeness, Deng et al. [11] have extended this result to disallow even an additive constant for a sufficiently small multiplicative factor. That is, for sufficiently small $c_1 > 0$ and for some $c_2 > 0$, there is no algorithm which can guarantee a solution within $(1 + c_1)optimum + c_2$ unless $NP = P$. This reduction method yields a simple proof of Jeroslow's original nonapproximability results [8]. This becomes quite obvious if one notices that the optimal value of the objective function in the above reduction is zero if and only if the 3SAT problem is satisfiable. Any multiplicative factor c will not change this fact. The problem with additive constant is a little tricky but can still be handled with ease.

3.4. Problems solvable in polynomial time. Due to the above negative results, positive results for the worst case complexity would only be possible for special classes of **BP**s. Liu and Spence have introduced a polynomial time algorithm for bilevel linear programming when there is a *constant* number of lower level control variables [21]. Deng et al. have presented a much simpler proof for this result [11] which also allows for an extension to a k-level linear programming problem when the total number of variables controlled by lower level linear programs is a constant [8].

The key idea of this result is the following. When the number of control variables of the upper level program is fixed, the lower level program will make its decision according to its linear program. We assume that the lower level program will choose among all its possible optimal solutions the best one for the upper level. This can be achieved at a basis solution. On the other hand, given this basis solution of the lower level program, first, we can check whether it is locally optimal, independent of the control variables of the upper level. Thus, we can solve the linear program of the upper level with the original constraints and this set of new linear constraints for this particular basis solution of the lower level program. If we could guess correctly which basis solution is the one of the optimum of the BP, then we are done. However, this is not difficult since there are at most a polynomial number of basis solutions for the lower level program. We can simply go through each of them.

A moderately interesting open problem concerns the complexity of the **BP** when the number of control variables of the upper level linear program is fixed as a constant. In another direction, suppose we allow a constant number of variables for both the upper and lower level programs but restrict the solution of the variables to be integers. Can we find a polynomial time algorithm using the techniques for solving integer programming of fixed dimension [20] ?

3.5. Hierarchy. For higher level hierarchies, in a very complicated proof, Jeroslow has shown that the optimal value of a $(k+1)$-level linear program is $\sum_k$-hard (and also Π_k-hard.) Again, similar to the bilevel case, Deng and Papadimitriou [**10**] have shown that the problem is $F\Delta_{k+1}^P$-complete. The same result holds even when we allow each linear program to have several sublinear programs with shared constraints under the assumption that the lower level agents reach a Nash Equilibrium point best for the upper level linear program (the focal point) as long as the total number of sublinear programs is a constant [**10**]. Thus, in this case, the complexity of a hierarchical structure is mainly determined by the number of its levels. We conjecture that this principle holds even when the lower level agents act in a different but definite way as discussed by Dubas et al., in [**12**].

4. Remarks

Although all the NP-hard complexity results characterize worst-case instances, they indicate the intractability of the hierarchical optimization problems. This explains the fact that in practice we can only solve to optimality problems of relatively small size. However, it is still possible that some heuristic methods may be successful for interesting special cases of hierarchical optimization problems.

From our complexity analysis, it seems that hierarchical structures are harder to manage than completely centralized systems. Then, what are the rationalities for hierarchical structures to exist? An answer to that question will require joint efforts by many scientists from diverse disciplines. In particular, answers to such questions may help us to understand the reason behind hierarchical structures in biology.

References

[1] G. Anandalingam and T.L. (Editors) Friesz, *Annals of Operations Research 34, Hierarchical Optimization*, Baltzer, 1992.

[2] E. Baer, A. Hiltner, and R. Morgan, *Biological and synthetic hierarchical composites*, Physics Today **46** (1992), 60–67.

[3] J. Berechman, *Highway-capacity utilization and investment in transportation corridors*, Environment and Planning **16A** (1984), 1475–1488.

[4] C. Blair, *The computational complexity of multi-level linear programs*, Annals of Operations Research **34** (1992), 13–19.

[5] J. Bracken and J.M. McGill, *Mathematical programs with optimization problems in the constraints*, Oper. Res. **21** (1973), 37–44.

[6] ———, *A method for solving mathematical programs with nonlinear programs in the constraints*, Oper. Res. **22** (1974), 1097–1101.

[7] J. M. Clark, *Economics and modern psychology*, Journal of Political Economy **26** (1918), 1–30.

[8] X. Deng, *Complexity issues in bilevel linear programming*, Multilevel Optimization: Algorithms and Applications, Kluwer Academic Publishers, 1997.

[9] X. Deng and C. H. Papadimitriou, *On the complexity of cooperative game solution concepts*, Mathematics of Operations Research **19** (1994), 257–266.

[10] ———, *Manuscript*, (1996).

[11] X. Deng, Q. Wang, and S. Wang, *On the complexity of linear bilevel programming*, Proceedings of the first International Symposium on Operations Research with Applications (1995), 205–212.

[12] T. Dubas, B. Klinz, and G.J. Woeginger, *The computational complexity of multi-level programming revisited*, Multilevel Optimization: Algorithms and Applications, Kluwer Academic Publishers, 1997.

[13] M. Garey and D. Johnson, *Computers and intractability a guide to the theory of np-completeness*, Freeman, San Francisco, 1979.

[14] Y.Y. Haimes, *Hierarchical analysis of water resource systems*, Mc Graw Hill, New York, 1977.
[15] P. Hansen, B. Jaumard, and G. Savard, *New branch-and-bound rules for linear bilevel programming*, SIAM J. Sci. Stat. Comput. **13** (1992), no. 5, 1194–1217.
[16] R. Horst and P.M. Pardalos, *Handbook of global optimization*, Kluwer Academic Publishers, 1995.
[17] R. Horst, P.M. Pardalos, and N.V. Thoai, *Introduction to global optimization*, Kluwer Academic Publishers, 1995.
[18] R.G. Jeroslow, *The polynomial hierarchy and a simple model for competitive analysis*, Mathematical Programming **32** (1985), 146–164.
[19] K.-I. Ko and C.-L. Lin, *On the complexity of min-max optimization problems and their approximation*, Minimax and Applications, Kluwer Academic Publishers, 1995, pp. 219–239.
[20] H.W. Lenstra, Jr, *Integer programming with a fixed number of variables*, Mathematics of Operations Research **8** (1983), 538–548.
[21] Y. Liu and T. H. Spencer, *Solving a bilevel linear program when the inner decision maker controls few variables*, European J. of Operations Research **81** (1995), 644–651.
[22] M. S. Mahoud, *Multilevel systems control and applications: A survey*, IEEE Trans. Systems, Man, and Cybernetics **7** (1977), 125–143.
[23] A. Migdalas and P.M. Pardalos, *Nonlinear bilevel problems with convex second level problem - heuristics and descent methods*, Operations Research and Its Applications, World Publishing Corporation, 1995, pp. 194–204.
[24] ———, *Hierarchical and bilevel programming*, Jounal of Global Optimization **8** (1996), no. 3.
[25] A. Migdalas, P.M. Pardalos, and P. Varbrand, *Multilevel optimization: Algorithms and applications*, Kluwer Academic Publishers, Boston, 1997.
[26] A. Neyman, *Bounded complexity justifies cooperation in the finitely repeated prisoners' dilemma*, Economics Letters **19** (1985), 227–229.
[27] C. H. Papadimitriou, *Computational complexity*, Addison-Welsley Publishing Company, Don Mills, Ontario, 1994.
[28] C.H. Papadimitriou and M. Yannakakis, *On complexity as bounded rationality*, Proceedings of the 26th ACM Symposium on the Theory of Computing, 1994, pp. 726–733.
[29] P.M. Pardalos, *Complexity in numerical optimization*, World Scientific, New York, 1993.
[30] P.M. Pardalos and G. Schnitger, *Checking local optimality in constrained quadratic programming is NP-hard*, Operations Research Letters **7** (1988), 33–35.
[31] Roy Radner, *Bounded rationality, indeterminacy, and the theory of the firm*, The Economic Journal **106** (1996), 1360–1373.
[32] H. Simon, *Theories of bounded rationality*, Decision and Organization, North Holland, 1992.
[33] K. Tarvainen and Haimes Y.Y., *Coordination of hierarchical multiobjective systems: Theory and methodology*, IEEE Trans. Systems, Man, and Cybernetics **7** (1977), 125–43.
[34] L. Vicente, G. Savard, and J. Judice, *Descent approaches for quadratic bilevel programming*, Journal of Optimization Theory and Applications **81** (1994), no. 2, 379–399.
[35] L.N. Vicente and P.H. Calamai, *Bilevel and multi level programming: A bibliographic review*, Journal of Global Optimization **5** (1994), 291–306.
[36] J.F.V. Vincent, *Structural biomaterials*, Princeton University Press, Princeton, 1990.

(P.M. Pardalos) CENTER FOR APPLIED OPTIMIZATION, DEPARTMENT OF INDUSTRIAL AND SYSTEMS ENGINEERING, UNIVERSITY OF FLORIDA, GAINESVILLE, FL 32611 USA

E-mail address, P.M. Pardalos: **pardalos@@ufl.edu**

(X. Deng) DEPARTMENT OF COMPUTER SCIENCE, YORK UNIVERSITY, NORTH YORK, ONTARIO, CANADA M3J 1P3

E-mail address, X. Deng: **deng@@cs.yorku.ca**

https://doi.org/10.1090/dimacs/037/14

DIMACS Series in Discrete Mathematics
and Theoretical Computer Science
Volume **37**, 1997

Nesticity

Pierre Hansen and Dominique de Werra

ABSTRACT. A hypergraph $H = (X, E)$ is *nested* if its edges are pairwise disjoint or included one into the other. The *nesticity* of H is defined as the smallest number of nested partial hypergraphs into which H can be partitioned. Determining the nesticity of H reduces to graph coloring. Hypergraphs of nesticity 2 are shown to be unimodular. A *prehypergraph* H' is obtained from a hypergraph H by specifying that some or all vertices from a given subset of each edge may be deleted. H' is thus a set of hypergraphs, called its *realizations*. Determining the nesticity of a prehypergraph H', i.e., the smallest nesticity of its realizations, reduces to satisfiability.

1. Nesticity of Hypergraphs

Let $X = \{x_1, x_2, \ldots, x_n\}$ denote a finite set, whose elements are called *vertices*. A *hypergraph* $H = (E_1, E_2, \ldots, E_m)$ on X (Berge [1]) is a finite family of non-empty subsets of X whose union is equal to X. These subsets are called *edges*. A hypergraph H is *nested* if its edges are pairwise disjoint or included one into the other:

$$(1) \qquad E_i \cap E_j = \emptyset \quad \text{or} \quad E_i \subseteq E_j \quad \text{or} \quad E_j \subseteq E_i \quad i, j = 1, 2, \ldots, m.$$

It is well-known that the clusters of entities obtained by any hierarchical clustering algorithm (see e.g., Gordon [8] for a recent survey) define a nested hypergraph. This hypergraph can be represented in various ways, e.g., by a tree, a dendrogram or an espalier [11].

If a hypergraph H is not nested, it is natural to ask into how many nested hypergraphs it may be decomposed. To make this question precise recall that a *partial hypergraph* H^* of H defined by the index set $J \subset \{1, 2, \ldots, m\}$ is the hypergraph $H^* = (E_j : j \in J)$. Thus, H^* is obtained from H by deleting the edges of index $j \in \{1, 2, \ldots, m\} \backslash J$ and the vertices belonging only to those edges.

The *nesticity* $N(H)$ of H is then defined as the smallest number of nested partial hypergraphs into which H can be partitioned. This concept was introduced (without a name) by Carroll and Corter [4].

1991 *Mathematics Subject Classification.* 05C78, 62H30.

The first author has been supported by ONR grant N00014-95-1-0917, FCAR grant 95-ER-1048 and NSERC grant GP0105574. Research done in part during a sabbatical leave of the first author at Département de Mathématiques, École Polytechnique Fédérale de Lausanne.
This paper is in final form and no version of it will be submitted elsewhere.

Application 1: *Multiple trees* ([4]).
Several models of additive cluster analysis seek how to explain observed pairwise similarities between entities of a given set by properties of subsets (or clusters) of them. To this effect, similarities between two entities are expressed as the sum of weights of the clusters to which both entities belong. Clusters and weights are simultaneously determined in order to minimize a loss function, such as the error sum-of-squares. The clusters obtained are overlapping and usually not nested. In order to represent results by multiple trees it is asked to partition the cluster set into the fewest possible nested hypergraphs.
Application 2: *Filiation of Manuscripts* (extension of a problem of Buneman[**3**]). Consider a set of manuscripts about the same cycle of legends. These manuscripts have been obtained through copying, with errors accumulating in the process. So, as long as a single manuscript is copied at a time, the complements of the sets of errors are nested. However, a manuscript may also be copied from two or more others, in which case some of the errors are eliminated. Manuscripts with nested complements of set of errors will in turn be obtained from this manuscript. To better understand the filiation between manuscripts one may consider a hypergraph $H = (E_1, E_2, \ldots, E_m)$ on X with X associated with the set of errors in all manuscripts, and $E_1, \ldots, E_m$ associated with the complements of the sets of errors in each of them. The nesticity of H will be equal to the number of source manuscripts plus the number of corrected ones.

To determine the nesticity $N(H)$ of a hypergraph H it is convenient to associate with H an *overlap graph* $G(H) = (\overline{X}, \overline{E})$ defined as follows ([4]): vertices $\overline{x}_i$ of $\overline{G}$ correspond to edges E_i of H and edge $\{\overline{x}_i, \overline{x}_j\}$ belongs to $\overline{E}$ if

$$E_i \cap E_j \neq \emptyset, \quad E_i \backslash E_j \neq \emptyset \quad \text{and} \quad E_j \backslash E_i \neq \emptyset, \tag{2}$$

i.e., if (1) does not hold for i and j. In that case, edges E_i and E_j cannot belong to the same nested partial hypergraph. Note that if H is a graph, $G(H)$ is its line-graph. A *coloring* of a graph $G = (X, E)$ is a partition of its vertex set X into *stable sets*, i.e., sets of vertices which are pairwise non adjacent. The *chromatic number* $\gamma(G)$ of G is the smallest number of stable sets in a coloring of G. The problem CHROMATIC NUMBER, in decision form, is defined as follows:

INPUT: Graph $G = (X, E)$, integer r.
QUESTION: Does G have a coloring in r colors ?

PROPOSITION 1. *Nesticity of hypergraphs reduces to* CHROMATIC NUMBER *as*

$$N(H) = \gamma\big(G(H)\big). \tag{3}$$

PROOF. This follows immediately from the fact that partitions of H into nested partial hypergraphs correspond to vertex colorings of $G(H)$. □

Carroll and Corter [**4**] give a result close to Proposition 1, in terms of the *nesting graph* $G'(H)$, which is the complementary graph of the overlap graph $G(H)$.

PROPOSITION 2. *Determining whether $N(H)$ is less than or equal to an integer r is NP-complete for $r \geq 3$.*

PROOF. Any graph $G = (\overline{X}, E)$ is the overlap graph of a hypergraph H, defined as follows: the vertex sets of H is equal to the vertex set $\overline{X} = \{\overline{x}_1, \overline{x}_2, \ldots, \overline{x}_n\}$ of G

plus a set of vertices $Y = \{y_1, \ldots, y_m\}$ associated with the edges of G. Then edges of H are equal to

$$\{\overline{x}_i \cup \{y_j : \text{ the edge associated with } y_j \text{ is incident with } \overline{x}_i \text{ in } G\}\}$$

for $i = 1, 2, \ldots, n$.

Clearly, no two edges of H are included one into the other. Moreover, they intersect if and only if they correspond to endpoints of an edge of G. Hence $G = G(H)$. The result then follows from CHROMATIC NUMBER being NP-complete for $\gamma(G) \geq 3$. □

Determining whether $N(H) = 2$ can be done in quadratic time: $G(H)$ can be constructed in such time by comparing pairwise lists of vertices in the edges. Then checking whether $\gamma(G(H)) = 2$ amounts to finding whether $G(H)$ contains an odd cycle or not which can be done in linear time in the size of the overlap graph, i.e., again in quadratic time in the size of the input.

Application 1 (continued). Carroll and Corter [4] consider the set of nine clusters for 20 instances of "sports" given in Table 1. These clusters are obtained from data of Smith *et al.* [15], using an extended version of MAPCLUS (Corter and Carroll [6]).

TABLE 1. Nine overlapping clusters from Carroll and Corter [4].

1	ping-pong billiard checkers
2	tennis volleyball ping-pong
3	hiking camping
4	football baseball basketball volleyball
5	canoeing swimming surfing skindiving
6	canoeing swimming skiing surfing skindiving horseback-riding hiking camping
7	canoeing archery horseback-riding hiking camping
8	boxing fencing archery
9	swimming skiing surfing skindiving

The corresponding overlap graph $G(H)$ is represented in Figure 1.

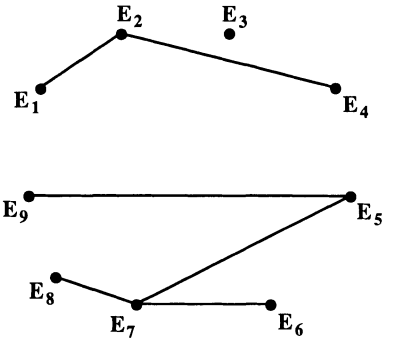


FIGURE 1. Overlap graph for the data of Table 1.

It is easily seen that the overlap graph $G(H)$ is 2-colorable. One bicoloration is $\{E_1, E_4, E_5, E_6, E_8\}, \{E_2, E_3, E_7, E_9\}$. Even if CHROMATIC NUMBER is NP-hard,

it is fairly easy to determine an optimal coloring in a graph with 50 to 100 vertices with a simple enumerative algorithm (e.g. Brelaz [2], Hansen and Delattre [9]). For larger graphs, enumerative methods using less immediate bounds can be used (e.g., Hertz and de Werra [12], and Dubois and de Werra [7]). It is also possible to modify backtracking rules in these algorithms in order to find all optimal colorings. A secondary criterion, as e.g. the total number of vertices in the nested partial hypergraphs, may then be taken into account.

Carroll and Corter [4] propose to find $N(H)$ by enumerating maximal cliques of the nesting graph $G'(H)$ and solving a covering problem. This last problem expresses that one seeks the minimum number of maximal cliques such that each vertex belongs to at least one of them. Such an approach is akin to the graph coloring algorithm of Christofides [5]. It is known that enumerative algorithms for $\gamma(G)$ which work directly on the vertices of G are more efficient.

We next show that hypergraphs of nesticity 2 have an important property.

PROPOSITION 3. *A hypergraph H with $N(H) \leq 2$ is unimodular.*

A hypergraph H is called *unimodular* if its edge-vertex incidence matrix is *totally unimodular*, i.e., if the determinant of any square submatrix is equal to -1, 0, or 1. It is known [1] that H is unimodular if and only if for every induced subhypergraph H^* of H (with edges $E_1^*, E_2^*, \ldots, E_m^*$) there exists a partition of the vertex set of H^* into 2 subsets I_1, I_2 such that for every edge E_i^* of H

$$-1 \leq |E_i^* \cap I_1| - |E_i^* \cap I_2| \leq +1$$

Such a partition is called an *equitable bicoloring.*

PROOF. We only have to show that any H with $N(H) \leq 2$ has an equitable bicoloring. Let F_1, F_2 be the two nested families of edges which cover the edge set of H. We may assume without loss of generality that each one contains the vertex set X of H as an edge and each vertex of H as an edge. We may represent them by oriented trees having the new edge X as root and the singletons (associated to the vertices of H) as leaves.

This construction is illustrated in Figure 2: each subset in F_1 (resp. F_2) is a node of a network N; we introduce arcs (E_i, E_j^-) (resp. $E_j^-, E_i)$) whenever $E_i, E_j \in F_1$ (resp. F_2), $E_i \subset E_j^-$ and there is no $E_h (h \neq i, j)$ such that $E_i \subset E_h \subset E_j$. We link the corresponding singletons in F_1 and in F_2 by arcs (x, x') with capacity 1 and lower bound of flow equal to zero.

For each arc (E_i, E_j) associated to F_1 (resp. (E_j, E_i) of F_2) we have $c(E_i, E_j) = \lceil |E_j|/2 \rceil, \ell(E_i, E_j) = \lfloor |E_j|/2 \rfloor$. Let s be the source of the network (it corresponds to the subset X in F_1 and t the sink (it corresponds to the subset X in F_2). Then there is a compatible flow from s to t in N; it is obtained by setting $f(x, x') = 1/2$ in each arc between singletons and by pushing this flow towards t and pulling it back towards s.

It is well known that there exists also an integral compatible flow in N (its values are 0 or 1 in the arcs (x, x')); it gives the required bipartition of the node set of H by setting $x \in I_1$ if $f(x, x') = 1$ and $x \in I_2$ if $f(x, x') = 0$. So H has an equitable bicoloring. □

REMARK 1. *By considering a hypergraph H consisting of edges $[a, x_i]$ for $i = 1, 2, \ldots, p$, we observe that a unimodular hypergraph H may have a nesticity $N(H)$ as large as we want: here $N(H) = p$.*

$$F_1 = (abcdefg, abc, ab, de, fg, a, b, c, d, e, f, g)$$
$$F_2 = (abcdefg, ade, ad, bg, cf, a, b, c, d, e, f, g)$$

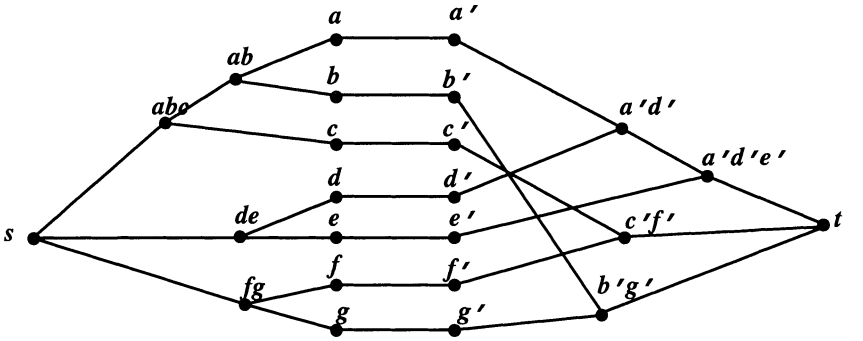


FIGURE 2. The network N associated with the nested families F_1, F_2.

2. Extension to Prehypergraphs

The second application mentionned above suggests an extension of the nesticity concept. Indeed, a rather strong underlying assumption is made: all manuscripts should be complete. If, as is likely to be the case, parts of some manuscripts are missing, it is impossible to know whether these missing parts contained or not errors appearing in corresponding parts of other manuscripts. One can then ask about the nesticity of a hypergraph in which the most favorable assumptions are made on the presence or absence of errors in missing parts.

To capture this situation we need to extend the concept of a hypergraph in order to allow some vertices to belong or not to some edges. Formally, a *prehypergraph* $H' = (E'_1, E''_1, E'_2, E''_2, \ldots, E'_m, E''_m)$, obtained from a hypergraph $H = (E_1, E_2, \ldots, E_m)$ is a family of hypergraphs $\overline{H} = (\overline{E}_1, \overline{E}_2, \ldots, \overline{E}_m)$, called its *realizations* and such that

$$\begin{aligned} E_i &= E'_i \cup E''_i, \quad E'_i \cap E''_i = \emptyset, \quad E'_i \neq \emptyset, \\ &\quad E'_i \subseteq \overline{E}_i \subseteq E'_i \cup E''_i \qquad i = 1, 2, \ldots, m. \end{aligned} \tag{4}$$

So, in any realization $\overline{H}$, edge $\overline{E}_i$ contains all vertices of E_i in E'_i, which is non-empty, and none, some or all of the vertices of E''_i, i.e., the remaining ones of E_i.

In some cases, it is easy to see that two edges $\overline{E}_i$ and $\overline{E}_j$ must overlap or cannot overlap. A sufficient condition for $\overline{E}_i$ and $\overline{E}_j$ to overlap is

$$E'_i \backslash E_j \neq \emptyset, \quad E'_j \backslash E_i \neq \emptyset, \quad \text{and} \quad E'_i \cap E'_j \neq \emptyset. \tag{5}$$

A sufficient condition for $\overline{E}_i$ and $\overline{E}_j$ not to overlap is

$$E_i \cap E_j = \emptyset. \tag{6}$$

In general conditions for absence of overlap will depend on the presence or absence of some vertices in $\overline{E}_i$ and $\overline{E}_j$. Let us then introduce variables

$$y_{ik} = \begin{cases} 1 & \text{if vertex } x_k \in \overline{E}_i, \\ 0 & \text{otherwise.} \end{cases} \tag{7}$$

There are three cases in which $\overline{E}_i$ and $\overline{E}_j$ do not overlap, which we study in turn. Let $\overline{E}_i'' = \overline{E}_i \cap E_i''$.

Case 1: $\overline{E}_i \cap \overline{E}_j = \emptyset$.

This condition breaks down into the four following ones, all to be satisfied:

(8) 1.a) $E_i' \cap E_j' = \emptyset$;

(9) 1.b) $E_i' \cap \overline{E}_j'' = \emptyset \iff \dot{\sum}_{k \mid x_k \in E_i' \cap E_j''} y_{jk} = 0$;

(10) 1.c) $\overline{E}_i'' \cap E_j' = \emptyset \iff \dot{\sum}_{k \mid x_k \in E_i'' \cap E_j'} y_{ik} = 0$;

(11) 1.d) $\overline{E}_i'' \cap \overline{E}_j'' = \emptyset \iff \dot{\sum}_{k \mid x_k \in E_i'' \cap E_j''} y_{ik} y_{jk} = 0$.

Case 2: $\overline{E}_i \subset \overline{E}_j$.

This condition breaks down into the three following ones, all to be satisfied:

(12) 2.a) $E_i' \backslash E_j = \emptyset$;

(13) 2.b) $(E_i' \cap E_j'') \backslash \overline{E}_j'' = \emptyset \iff \dot{\sum}_{k \mid x_k \in E_i' \cap E_j''} \overline{y}_{jk} = 0$

where $\overline{y}_{jk} = 1 - y_{jk}$;

(14) 2.c) $(\overline{E}_i'' \cap E_j'') \backslash \overline{E}_j = \emptyset \iff \dot{\sum}_{k \mid x_k \in E_i'' \cap E_j''} y_{ik} \overline{y}_{jk} = 0$.

Case 3: $\overline{E}_j \subset \overline{E}_i$.

This condition is similar to Case 2. Conditions are obtained by permuting indices i and j in (12)–(14).

The sums $\dot{\sum}$ appearing in conditions (9)–(11), (13), (14) are boolean ones, i.e. defined by

$$\begin{aligned} 0 \dot{+} 0 &= 0, \\ 1 \dot{+} 0 &= 1, \\ 0 \dot{+} 1 &= 1, \\ 1 \dot{+} 1 &= 1. \end{aligned} \tag{15}$$

Let us then introduce variables z_{ij}^1, z_{ij}^2 and z_{ij}^3 which may be equal to 1 if the conditions of case 1, case 2 and case 3 respectively are satisfied and otherwise are equal to 0. To ensure this, we impose the constraints:

$$z_{ij}^1 \Big(a_{ij} \dot{+} \dot{\sum}_{k \mid x_k \in E_i' \cap E_j''} y_{jk} \dot{+} \dot{\sum}_{k \mid x_k \in E_i'' \cap E_j'} y_{ik} \dot{+} \dot{\sum}_{k \mid x_k \in E_i'' \cap E_j''} y_{ik} y_{jk} \Big) = 0 \tag{16}$$

where

$$a_{ij} = \begin{cases} 1 & \text{if } E'_i \cap E'_j \neq \emptyset, \\ 0 & \text{otherwise;} \end{cases} \tag{17}$$

$$z_{ij}^2 \Big(b_{ij} + \dot{\sum_{k|x_k \in E'_i \cap E''_j}} \overline{y}_{jk} \dot{+} \dot{\sum_{k|x_k \in E''_i \cap E''_j}} y_{ik} \overline{y}_{jk} \Big) = 0 \tag{18}$$

where

$$b_{ij} = \begin{cases} 1 & \text{if } E'_i \backslash E_j \neq \emptyset \\ 0 & \text{otherwise;} \end{cases} \tag{19}$$

$$z_{ij}^3 \Big(c_{ij} \dot{+} \dot{\sum_{k|x_k \in E''_i \cap E_j}} \overline{y}_{ik} \dot{+} \dot{\sum_{k|x_k \in E''_i \cap E''_j}} \overline{y}_{ik} y_{jk} \Big) = 0 \tag{20}$$

where

$$c_{ij} = \begin{cases} 1 & \text{if } E'_j \backslash E_i \neq \emptyset, \\ 0 & \text{otherwise.} \end{cases} \tag{21}$$

Let us further introduce variables z_{ij}, which may be equal to 1 if $\overline{E}_i$ and $\overline{E}_j$ do not overlap, and equal to 0 otherwise: This is expressed by the conditions:

$$z_{ij} \quad \overline{z}_{ij}^1 \quad \overline{z}_{ij}^2 \quad \overline{z}_{ij}^3 = 0, \tag{22}$$

i.e., z_{ij} may be equal to 1 if one of z_{ij}^1, z_{ij}^2 and z_{ij}^3 is equal to 1.

Note that if condition (5) holds $z_{ij} = 0$, if condition (6) holds z_{ij} may be set equal to 1; if a_{ij}, b_{ij} or c_{ij} is equal to 1, z_{ij}^1, z_{ij}^2 or z_{ij}^3 respectively is equal to 0. These observations may be used to simplify the list of conditions for having $z_{ij} = 1$.

We now turn to an expression of nesticity for the prehypergraph H' itself. Assume an upper bound U on this nesticity is known (such a bound may be obtained by computing the nesticity of H). Then a dichotomous search on the range $[1, U]$ is performed. For a given value L in this range, we check for the existence of a realization $\overline{H}$ of H with nesticity L. To this effect let

$$t_{i\ell} = \begin{cases} 1 & \text{if edge } \overline{E}_i \text{ is in the } \ell^{th} \text{ nested partial hypergraph,} \\ 0 & \text{otherwise.} \end{cases}$$

Consider the set of equations

$$\overline{z}_{ij} \Big(\dot{\sum_{\ell=1}^{L}} t_{i\ell} t_{j\ell} \Big) = 0 \qquad i, j = 1, 2, \ldots, m, \quad \ell = 1, 2, \ldots, L \tag{23}$$

$$\prod_{\ell=1}^{L} \overline{t}_{i\ell} = 0 \qquad i = 1, 2, \ldots, m \tag{24}$$

and (16) (18) (20) (22) for all $i, j = 1, 2, \ldots, m$.

Equation (23) expresses that edges $\overline{E}_i$ and $\overline{E}_j$ may belong to the same nested hypergraph only if $z_{ij} = 1$, which happens only when one of the conditions in the parenthesis of (16) (18) or (20) is satisfied, i.e., equal to 0, or, in other words, in one of the three cases in which $\overline{E}_i$ and $\overline{E}_j$ do not overlap. Equation (24) implies that every edge $\overline{E}_i$ is assigned to one at least of the L nested partial hypergraphs

of the partition considered. If an edge is assigned to several partial hypergraphs it is straightforward to obtain a solution in which this does not happen by keeping only first assignments.

Conditions (16) (18) (20) (22) (23) and (24) are equal, after a few multiplications, to boolean sums of products of boolean variables. Gathering them gives a boolean formula in disjunctive normal form. We have then proved:

PROPOSITION 4. *Nesticity of prehypergraphs reduces to* SATISFIABILITY.

From Proposition 2, determining that the nesticity of a prehypergraph is equal to r is NP-complete for $r \geq 3$. The complexity of the case $r = 2$ is open.

While the number of variables and constraints used in the reduction given above is substantial, recent progress in algorithms for SATISFIABILITY (see e.g., Hansen and Jaumard [**10**], Selman, Levesque and Mitchell [**14**] Jaumard, Minhea and Desrosiers [**13**],) makes it likely that nesticity can be determined for moderate size partially defined hypergraphs in reasonable time.

References

[1] Berge, C., *Hypergraphes*, Paris, Dunod (1987).

[2] Brelaz, D. New Methods to Color the Vertices of a Graph, *Communications of the ACM* 22 (1979) 251–256.

[3] Buneman, P., Filiation of Manuscripts, *Mathematics in the Archaeological and Historical Sciences*, Edinburgh: Edinburgh University Press (1971) 387–395.

[4] Carroll, J.D., and Corter, J.E., A Graph-theoretic Method for Organizing Overlapping Clusters into Trees, Multiple Trees or Extended Trees, *Journal of Classification* 12 (1995) 283–313.

[5] Christofides, N., An Algorithm for the Chromatic Number of a Graph, *The Computer Journal* 14 (1971) 38–39.

[6] Corter, J.E., and Carroll, J.D., Automatic Fitting of Complementary Clusters in MAPCLUS, Unpublished Manuscript, Columbia University (1993).

[7] Dubois, N., and de Werra, D., EPCOT: An Efficient Procedure for Coloring Optimally with Tabu Search, *Computers and Mathematics with Applications* 25 (1993) 35–45.

[8] Gordon, A.D., A Review of Hierarchical Classification, *Journal of the Royal Satistical Society* A 150 (1987) Part 2, 119-137.

[9] Hansen, P., and Delattre, M., Complete-link Cluster Analysis by Graph Coloring, *Journal of the American Statistical Association* 73 (1978) 397–403.

[10] Hansen, P., and Jaumard, B., Algorithms for the Maximum Satisfiability Problem, *Computing* 44 (1990) 279–303.

[11] Hansen, P., Jaumard, B., and Simeone, B., Espaliers, a Generalization of Dendrograms, *Journal of Classification* 13 (1996) 107–127.

[12] Hertz, A., and de Werra, D., Using Tabu Search Techniques for Graph Coloring, *Computing* 29 (1987) 345–351.

[13] Jaumard, B., Stan, and M., Desrosiers, J., Tabu Search and Quadratic Relaxation for the Satisfiability Problem, in Johnson, D.S. and Trick, M. (editors) *Cliques, Coloring and Satisfiability, DIMACS Series in Discrete Mathematics and Theoretical Computer Science* 26 (1996) 457–477.

[14] Selman, B., Levesque, H.J., and Mitchell, D.G., A New Method for Solving Hard Satisfiability Problems, *Proceedings AAAI-92*, San-José, CA (1992) 440–446.

[15] Smith, E.E., Rips, L.J., Sholen, E.J., Rosch, E., and Mervis, C.G., Unpublished data (cited in [3]), (1975).

GERAD AND ÉCOLE DES HAUTES ÉTUDES COMMERCIALES, MONTRÉAL
E-mail address: pierreh@crt.umontreal.ca

DÉPARTEMENT DE MATHÉMATIQUES, ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE
E-mail address: dewerra@dma.epfl.ch

https://doi.org/10.1090/dimacs/037/15

DIMACS Series in Discrete Mathematics
and Theoretical Computer Science
Volume **37**, 1997

Phylogeny Graphs of Arbitrary Digraphs

Fred S. Roberts and Li Sheng

ABSTRACT. If $D = (V, A)$ is a digraph, its *competition graph* $C(D)$ is the graph $G = (V, E)$ with the same vertex set and an edge $\{x, y\}$ in E for $x \neq y$ iff there is a vertex a in V so that arcs (x, a) and (y, a) are in D. Its *phylogeny graph* is the graph $G = (V, E)$ with an edge between $x \neq y$ in E iff either (x, a) and (y, a) are in A for some a in V or (x, y) is in A or (y, x) is in A. Problems of ecology gave rise to the first concept and problems of phylogenetic tree reconstruction to the second. Roberts and Sheng [1996] noted the parallel between these two concepts and proved a variety of results about phylogeny graphs analogous to useful and well-known theorems about competition graphs. In that paper, D is usually assumed to be acyclic, as is common in the ecological applications and follows from the usual assumptions in phylogenetic tree reconstruction. We continue the study of the analogies between competition graphs and phylogeny graphs, emphasizing the case where D is an arbitrary digraph, not necessarily acyclic.

1. Introduction

In this paper, we consider the analogy between two graph-theoretical concepts motivated by problems of the biological sciences. Problems of ecology stimulated the development of the concept of competition graph in 1968, and have led to a large literature that is summarized in the articles by Kim [1993], Lundgren [1989], and Roberts [1997]. Motivated by problems of phylogenetic tree reconstruction, Roberts and Sheng [1996] introduced an analogous concept of phylogeny graph, and obtained a variety of results about phylogeny graphs that paralleled results in the theory of competition graphs. In this paper, we present several more parallel results.

We use the notation and terminology of Roberts [1976,1984]. If $D = (V, A)$ is a digraph, its *competition graph* $C(D)$ is the graph $G = (V, E)$ with the same vertex set and an edge $\{x, y\}$ in E for $x \neq y$ iff there is a vertex a in V so that arcs (x, a) and (y, a) are in D. Cohen [1968] introduced competition graphs when he was thinking of D as representing a food web in an ecosystem, with vertices the species in the system and an arc from u to v meaning that u preys on v. Then x and y are adjacent in $C(D)$ iff they have a common prey. Competition graphs also have applications in communications, coding, scheduling, and modeling of complex energy and economic systems (see Raychaudhuri and Roberts [1985] and Roberts [1997]).

1991 *Mathematics Subject Classification.* Primary 05C90, 92D15, 92D40, 05C20.

The authors gratefully acknowledge the support of the National Science Foundation under grant number BIR 94-12594 to Rutgers University.

If $D = (V, A)$ is a digraph, its *phylogeny graph* is the graph $G = (V, E)$ with an edge between $x \neq y$ in E iff either (x, a) and (y, a) are in A for some a in V or (x, y) is in A or (y, x) is in A. This concept was motivated by phylogeny in the following way. Let us measure the distance $D(x, y)$ between two species in a phylogenetic (evolutionary) tree by the distance to the closest common ancestor, *i.e.*, the minimum d so that x and y have a common ancestor reachable from x and y by paths of length at most d. Suppose we are given a set S of species and some measure of the similarity between two species. We wish to represent the species as vertices of a phylogenetic tree so that species x and y are more similar than species u and v iff $D(x, y) < D(u, v)$. In the special case where similarities are all either 0 or 1, we then ask that x and y have similarity 0 iff $D(x, y) = 1$ and similarity 1 iff $D(x, y) = 2$. A natural modification of this is to ask that x and y have similarity 0 iff $D(x, y) = 1$ and similarity 1 iff $D(x, y) > 1$. In this case, we seek a phylogenetic tree so that $x \neq y$ have similarity 0 iff either (x, a) and (y, a) are arcs of the tree for some a, or (x, y) or (y, x) is an arc of the tree. This gives rise to the notion of phylogeny graph. While the motivation is somewhat distant from the practical problems of phylogenetic tree reconstruction, nevertheless it gives rise to an interesting analogue of competition graph and it can be hoped that exploiting the analogy between competition graphs and phylogeny graphs can eventually lead to interesting conclusions about phylogenetic tree reconstruction problems.

If D is a digraph without loops and D' is the corresponding digraph with a loop added to each vertex, then it is easy to see that the phylogeny graph of D is the competition graph of D'. The authors thank F.R. McMorris for this observation. In the early literature of competition graphs, it was assumed, because of the ecological motivation, that the digraph D was acyclic and had no loops. However, eventually these assumptions were weakened in part because of non-biological applications and in part because not all food webs are acyclic or loop-free. It is the weakened cases that we study here. Of course, even allowing D to be an arbitrary acyclic digraph rather than a rooted tree oriented toward the root, as is common in phylogeny, already departs from the phylogeny application, so weakening the properties of D departs even further from this application. The results we present here are relatively straightforward. However, they introduce the subject of phylogeny graphs of arbitrary digraphs, a subject which we hope will eventually give rise to results as interesting as those in the literature of competition graphs and which might, eventually, be exploited to say something interesting about phylogenetic tree reconstruction or, what is more likely, about some of the many applications of competition graphs.

2. The Acyclic Case

A major problem in the theory of competition graphs is to recognize when a graph is the competition graph of some digraph with certain properties. For the case of acyclic digraphs, the problem is NP-complete (Opsut [1982]). (The same is true for phylogeny graphs, as shown by Roberts and Sheng [1996].) However, we have the following characterization theorem. Here, an ECC or *edge clique covering* of graph G is a collection of cliques (not necessarily maximal) of G covering all edges of G.

THEOREM 1. **(Dutton & Brigham [1983], Lundgren & Maybee [1983]).** *A graph G with n vertices is a competition graph of some acyclic digraph if and*

only if the vertices of G can be labeled v_1, v_2, ..., v_n so that G has an ECC $\{C_1, C_2, ..., C_n\}$ such that if $v_i \in C_j$, then $i > j$.

Analogously, for phylogeny graphs, we have:

THEOREM 2. **(Roberts and Sheng [1996]).** *A graph G with n vertices is a phylogeny graph of some acyclic digraph if and only if the vertices of G can be labeled v_1, v_2, ..., v_n so that G has an ECC $\{C_1, C_2, ..., C_{n-1}\}$ such that $v_j \in C_j$ and if $v_i \in C_j$, then $i \geq j$.*

3. Arbitrary Digraphs

Roberts and Steif [1983] gave results for competition graphs of arbitrary digraphs (cycles allowed) having no loops and Dutton and Brigham [1983] did the same for competition graphs of arbitrary digraphs with loops allowed. Their results are as follows.

THEOREM 3. **(Dutton and Brigham [1983]).** *A graph G with n vertices is the competition graph of an arbitrary digraph with loops allowed iff G has an ECC of at most n cliques.*

THEOREM 4. **(Roberts and Steif [1983]).** *A graph G with n vertices is the competition graph of an arbitrary digraph without loops iff G has an ECC of at most n cliques and G is not K_2.*

We first give an analogue of Theorems 3 and 4 for phylogeny graphs. Note that if $G = P(D)$, then (x, x) is a loop in D iff $\{x, x\}$ is a loop in G. So, if we don't allow loops in G, we cannot allow them in D.

THEOREM 5. *A graph G (without loops) with n vertices is the phylogeny graph of an arbitrary digraph (without loops) iff G has an ECC of at most n cliques.*

Proof. Suppose that $G = P(D)$ for $D = (V, A)$ an arbitrary digraph without loops, and label the vertices of D as v_1, v_2, ..., v_n. Let $C_j = \{v_i : (v_i, v_j) \in A\} \cup \{v_j\}$, $j = 1, 2, ..., n$. Then it is easy to show that $\{C_1, C_2, ..., C_n\}$ is an ECC for G.

Conversely, suppose that $\{C_1, C_2, ..., C_t\}$ is an ECC for G with $t \leq n$. For $i = 1, ..., n$, choose v_i in C_i and define $D = (V, A)$ by letting A consist of all arcs of the form (x, v_j) for $x \neq v_j$, $x \in C_j$. Then it is easy to see that $G = P(D)$. □

Note: The first direction in the proof can be replaced by the following argument. Suppose $G = P(D)$. Then $G = C(D')$ where D' is defined in the last paragraph of Section 1. By Theorem 3, G has an ECC of at most n cliques.

4. Competition Numbers and Phylogeny Numbers

In attempting to understand what graphs arose as competition graphs of acyclic digraphs, Roberts [1978] noted that if G is any graph, then G plus sufficiently many isolated vertices is the competition graph of an acyclic digraph. He called the smallest r so that G plus r isolated vertices is the competition graph of an acyclic digraph the *competition number* $k(G)$ of G. Characterization of competition graphs is equivalent to computation of competition numbers. There is a large literature about competition numbers, and here we simply state one result which we intend to parallel for phylogeny graphs.

The following result (stated as corrected by Kim [1988]), gives an upper bound on the competition number.

THEOREM 6. **(Lundgren and Maybee [1983])**. *If G is a graph of n vertices and $m \leq n$, then $k(G) \leq m$ iff the vertices of G can be labeled $v_1, v_2, \ldots, v_n$ so that G has an ECC $\{C_1, C_2, \ldots, C_{n+m-2}\}$ such that if $v_i \in C_j$, then $i \geq j - m + 1$.*

Any acyclic digraph D for which G is a generated subgraph of $P(D)$ and such that D has no arcs from vertices outside of G to G is called a *phylogeny digraph* for G. (Note that we do not restrict ourselves to oriented trees or insist that the species in G be leaves of the tree, as is common in phylogenetic tree reconstruction.) Roberts and Sheng [1996] define the *phylogeny number* $p(G)$ of G to be the smallest r so that G has a phylogeny digraph D with $|V(D)| - |V(G)| = r$. They show this is well-defined and note that, unlike the situation with competition number, there can be edges in $P(D)$ between vertices of G and vertices of D not in G.

The next result is analogous to the Lundgren-Maybee Theorem.

THEOREM 7. *If G is a graph of n vertices and $m \leq n$, then $p(G) \leq m$ iff the vertices of G can be labeled $v_1, v_2, \ldots, v_n$ so that G has an ECC $\{C_1, C_2, \ldots, C_{n+m-1}\}$ such that (i) $v_j \in C_{j+m}$ for $j = 1, 2, \ldots, n-1$ and such that (ii) if $v_i \in C_{j+m}$, $j \geq 1$, then $i \geq j$.*

Proof. Suppose $p(G) \leq m$ and D is an acyclic digraph such that G is a generated subgraph of $P(D)$, D has vertices $a_1, a_2, \ldots, a_m$ in addition to the vertices of G, and D has no arcs from vertices a_i to vertices of G. By acyclicity of D, the vertices of G can be labeled $v_1, v_2, \ldots, v_n$ so that $(v_i, v_j) \in A(D)$ implies $i > j$. Define

$$C_j = \{v_i : (v_i, a_j) \in A(D)\},\ j = 1, 2, \ldots, m$$

and

$$C_{j+m} = \{v_i : (v_i, v_j) \in A(D)\} \cup \{v_j\},\ j = 1, 2, \ldots, n-1.$$

Since G is a generated subgraph of $P(D)$, each C_k is a clique and the C_k, $k = 1, 2, \ldots, m + n - 1$, form an ECC for G. By construction, conditions (i) and (ii) hold.

Conversely, suppose that we are given a vertex labeling and an ECC satisfying the conditions of the theorem. Let $a_1, a_2, \ldots, a_m$ be new vertices and define a digraph D on the vertex set $V(G) \cup \{a_1, a_2, \ldots, a_m\}$ as follows: If $v_i \in C_j$, $j = 1, 2, \ldots, m$, then let $(v_i, a_j) \in A(D)$; if $v_i \in C_{j+m}$, $j = 1, 2, \ldots, n-1$, then let $(v_i, v_j) \in A(D)$ as long as $i \neq j$. The digraph is acyclic because if $(v_i, v_j) \in A(D)$, then $v_i \in C_{j+m}$, $j \geq 1$, and $i \neq j$, so by (ii), $i > j$. It is straightforward to verify that G is a generated subgraph of $P(D)$. (Condition (i) is used to show that if there is an arc from v_i to v_j in D, v_i and v_j are both in C_{j+m} and so are adjacent in G.) □

References

[1] Cohen, J.E., Interval Graphs and Food Webs: A Finding and a Problem, *RAND Corporation Document 17696-PR*, Santa Monica, CA, 1968.

[2] Dutton, R.D., and Brigham, R.C., A Characterization of Competition Graphs, *Discrete Applied Math.*, **6** (1983), 315-317.

[3] Kim, S.-R., *Competition Graphs and Scientific Laws for Food Webs and Other Systems*, Ph.D. Thesis, Department of Mathematics, Rutgers University, New Brunswick, NJ, October 1988.

[4] Kim, S.-R., The Competition Number and its Variants, in J. Gimbel, J.W. Kennedy, and L.V. Quintas, eds., *Quo Vadis Graph Theory?, Annals of Discrete Mathematics*, Vol. **55**, 1993, 313-325.

[5] Lundgren, J.R., Food Webs, Competition Graphs, Competition-Common Enemy Graphs, and Niche Graphs, in F.S. Roberts (ed.), *Applications of Combinatorics and Graph Theory in the*

Biological and Social Sciences, Vol. 17 of IMA Volumes in Mathematics and its Applications, Springer-Verlag, New York, 1989, 221-243.

[6] Lundgren, J.R., and Maybee, J.S., A Characterization of Graphs of Competition Number m, *Discrete Applied Math.*, **6** (1983), 319-322.

[7] Opsut, R.J., On the Computation of the Competition Number of a Graph, *SIAM J. Alg. Discr. Meth.*, **3** (1982), 420-428.

[8] Raychaudhuri, A., and Roberts, F.S., Generalized Competition Graphs and their Applications, in P. Brucker and R. Pauly (eds.), *Methods of Operations Research*, **49**, Anton Hain, Konigstein, West Germany, 1985, 295-311.

[9] Roberts, F.S., *Discrete Mathematical Models, with Applications to Social, Biological, and Environmental Problems*, Prentice-Hall, Englewood Cliffs, N.J. 1976.

[10] Roberts, F.S., Food Webs, Competition Graphs, and the Boxicity of Ecological Phase Space, in Y. Alavi and D. Lick (eds.), *Theory and Applications of Graphs*, Springer-Verlag, New York, 1978, 477-490.

[11] Roberts, F.S., *Applied Combinatorics*, Prentice-Hall, Englewood Cliffs, N.J., 1984.

[12] Roberts, F.S., Competition Graphs and Phylogeny Graphs, preprint, Department of Mathematics, Rutgers University, New Brunswick, NJ, 1997.

[13] Roberts, F.S., and Sheng, L., Phylogeny Numbers, preprint, Department of Mathematics and RUTCOR-Rutgers Center for Operations Research, Rutgers University, New Brunswick, NJ, 1996.

[14] Roberts, F.S., and Steif, J.E., A Characterization of Competition Graphs of Arbitrary Digraphs, *Discrete Applied Math.*, **6** (1983), 323-326.

Department of Mathematics, RUTCOR, and DIMACS, Rutgers University, New Brunswick, NJ, USA.

E-mail address: froberts@dimacs.rutgers.edu

RUTCOR – Rutgers Center for Operations Research, Rutgers University, New Brunswick, NJ, USA.

E-mail address: lisheng@dimacs.rutgers.edu

This page intentionally left blank

https://doi.org/10.1090/dimacs/037/16

DIMACS Series in Discrete Mathematics
and Theoretical Computer Science
Volume **37**, 1997

Agreement metrics for trees revisited

Ewa Kubicka, Grzegorz Kubicki and F. R. McMorris

ABSTRACT. Agreement metrics based on the agreement subtree were previously proposed for labeled binary trees. We continue the analysis of these metrics, introduce a modification, and study the structure of geodesics.

1. Introduction

It is well established that developing rigorous methods for comparing trees is important in systematic biology and classification theory. For example, in [3] and [4], two metrics were introduced on the set of all leaf-labeled binary trees with n leaves. Both were based on the notion of the agreement subtree of two trees, which is a subtree obtained by pruning leaves from the two trees. In the present paper, we propose some modifications of these metrics based on our study of the intervals between trees.

2. Definitions and notation.

Let S be a set with n elements. A *labeled binary tree* on S is a tree whose leaves are in one-to-one and onto correspondence with elements of S and whose internal vertices are unlabeled with degree three. $\mathcal{T}_n$ will denote the set of all labeled binary trees on S. We usually will just refer to an element $T \in \mathcal{T}_n$ as a "tree".

For $T \in \mathcal{T}_n$ and $X \subseteq S$, *pruning* the leaves in X means removing X from T and suppressing the vertices of degree 2 that are formed by this removal. This process is illustrated in Figure 1.

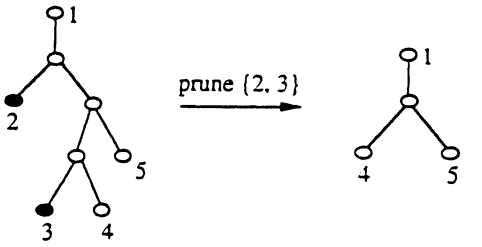


Figure 1

1991 *Mathematics Subject Classification.* Primary 92B10; Secondary 51K99.
The third author was supported by ONR grant N00014-95-1-0109.

A *common pruned subtree* for $T_1, T_2 \in \mathcal{T}_n$ is a tree that can be obtained from both T_1 and T_2 by pruning, and an *agreement subtree* is a common pruned subtree with the maximum number of leaves. $\mathcal{A}(T_1, T_2)$ will denote the set of all agreement subtrees for the two trees T_1 and T_2 and let $\#(T_1, T_2)$ denote the number of leaves in an agreement subtree.

3. The simple metric and intervals

We now recall the most straightforward properties of the metrics on $\mathcal{T}_n$ that are based on the agreement subtree. The function $d: \mathcal{T}_n \times \mathcal{T}_n \to \mathbf{R}$ defined by

$$d(T_1, T_2) = n - \#(T_1, T_2) \tag{1}$$

for all $T_1, T_2 \in \mathcal{T}_n$ is clearly a metric on $\mathcal{T}_n$. This metric is appealing since it simply counts the number of leaves which have to be pruned from both trees to obtain a common substructure, and there are polynomial algorithms to compute d ([1], [2], [3], and [5]).

The interval determined by T_1 and T_2 is the following family of trees:

$$\mathcal{I}(T_1, T_2) = \{T \in \mathcal{T}_n : d(T_1, T) + d(T, T_2) = d(T_1, T_2)\}. \tag{2}$$

Before we characterize the interval for two arbitrary trees of $\mathcal{T}_n$, we need some terminology. Suppose that A is an agreement subtree of T_1 and T_2 obtained by pruning $k+m$ leaves $u_1, u_2, \ldots, u_k, v_1, v_2, \ldots, v_m$. The *tree obtained from A by regrafting* $\{u_1, u_2, \ldots, u_k\}$ *according to* T_1 is the tree

$$A \oplus_{T_1} \{u_1, u_2, \ldots, u_k\} \cong T_1 - v_1 - v_2 - \ldots - v_m. \tag{3}$$

Similarly, *the tree obtained from A by regrafting* $\{v_1, v_2, \ldots, v_m\}$ *according to* T_2 is defined by

$$A \oplus_{T_2} \{v_1, v_2, \ldots, v_m\} \cong T_2 - u_1 - u_2 - \ldots - u_k. \tag{4}$$

The set of trees obtained from A by regrafting $\{u_1, u_2, \ldots, u_k\}$ according to T_1 and $\{v_1, v_2, \ldots, v_m\}$ according to T_2 is defined as

$$\begin{aligned} &\{u_1, u_2, \ldots, u_k\} \oplus_{T_1} A \oplus_{T_2} \{v_1, v_2, \ldots, v_m\} \\ &\quad = \{T \in \mathcal{T}_n : T - v_1 \ldots - v_m \cong A \oplus_{T_1} \{u_1, u_2, \ldots, u_k\} \\ &\quad \text{and } T - u_1 - \ldots - u_k \cong A \oplus_{T_2} \{v_1, v_2, \ldots, v_m\}\}. \end{aligned} \tag{5}$$

In general, by such double regrafting, we obtain a family of trees, not a single tree.

Theorem 1. *Let T_1 and T_2 be arbitrary trees in $\mathcal{T}_n$. A tree T is in $\mathcal{S}(T_1, T_2)$ if and only if there exists an agreement subtree A of T_1 and T_2 and a bi-partition $\{u_1, u_2, \ldots, u_k\}, \{v_1, v_2, \ldots, v_m\}$ of the pruned vertices such that*

$$T \in \{u_1, u_2, \ldots, u_k\} \oplus_{T_1} A \oplus_{T_2} \{v_1, v_2, \ldots, v_m\}. \tag{6}$$

PROOF. Suppose first that

$$T \in \{u_1, u_2, \ldots, u_k\} \oplus_{T_1} A \oplus_{T_2} \{v_1, v_2, \ldots, v_m\}. \tag{7}$$

Then $d(T, T_1) = m$ and $d(T, T_2) = k$. Of course, $d(T_1, T_2) = k + m$ and, therefore, $T \in \mathcal{S}(T_1\ T_2)$.

Conversely, suppose that $d(T_1, T_2) = d(T_1, T) + d(T, T_2) = k + m$ and $d(T_1, T) = m$. Then $d(T_2, T) = k$. There exist $A_1 \in \mathcal{A}(T_1, T)$ of size $n - m$ and $A_2 \in \mathcal{A}(T_2, T)$ of size $n - k$. The tree A_1 is obtained from T_1 (or from T) by pruning m leaves, say $v_1, v_2, \ldots, v_m$. The tree A_2 is obtained from T_2 (or from T) by pruning k leaves, say $u_1, u_2, \ldots, u_k$. Let A be the tree obtained by pruning $k + m$ leaves $u_1, u_2, \ldots, u_k, v_1, v_2, \ldots, v_m$ from T (see Figure 2). Of course, all leaves u_i's and v_j's are distinct because otherwise we could remove less than $k + m$ leaves to produce an agreement subtree for T_1 and T_2.

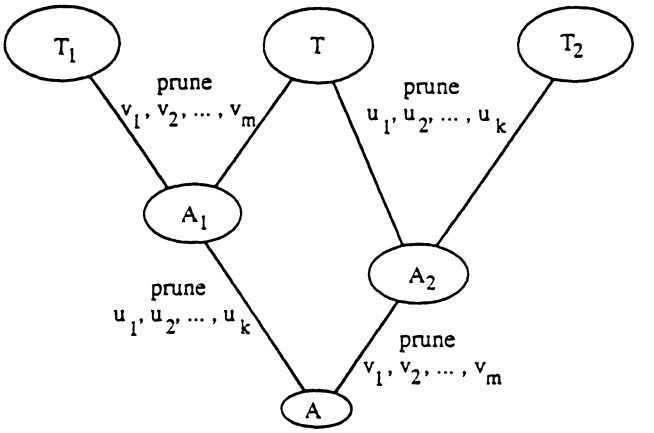


Figure 2

We will justify that $A \in \mathcal{A}(T_1, T_2)$. Of course, A is a common pruned subtree of T_1 and T_2. A is a maximum common pruned subtree of T_1 and T_2, because its size is $n - (k + m)$ and $d(T_1, T_2) = k + m$.

Notice that

$$A_1 \equiv A \oplus_{T_1} \{u_1, u_2, \ldots, u_k\} \quad \text{and} \tag{8}$$
$$A_2 \equiv A \oplus_{T_2} \{v_1, v_2, \ldots, v_m\}.$$

Therefore, $T \in \{u_1, u_2, \ldots, u_k\} \oplus_{T_1} A \oplus_{T_2} \{v_1, v_2, \ldots, v_m\}$. ■

For illustration, in Figure 3, the portion of the interval $\mathcal{I}(T_1, T_2)$ is presented for given trees $T_1, T_2 \in \mathcal{T}_8$ and for one of their agreement subtrees. The three trees in the second row represent the family $\{c, f\} \oplus_{T_1} A \oplus_{T_2} \{d\}$.

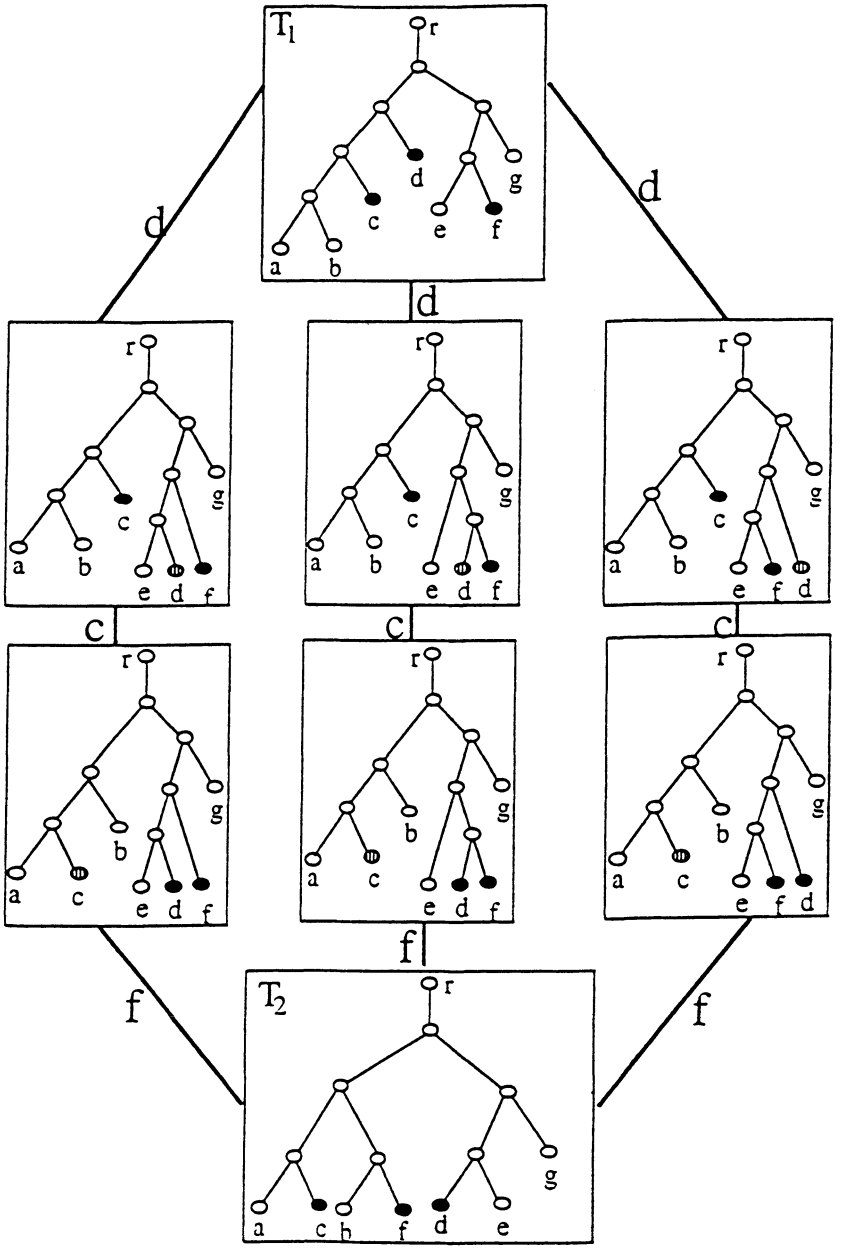


Figure 3
Part of $\mathcal{I}(T_1, T_2)$ for an agreement subtree A determined by white leaves

The diagram in Figure 4 gives the structure of the interval $\mathcal{I}(T_1, T_2)$ for a given agreement subtree A. Two trees in this diagram are joined by an edge if the distance between them is 1. In such a case it is possible to prune one leaf from one tree above and regraft it according to T_2 to get a tree below.

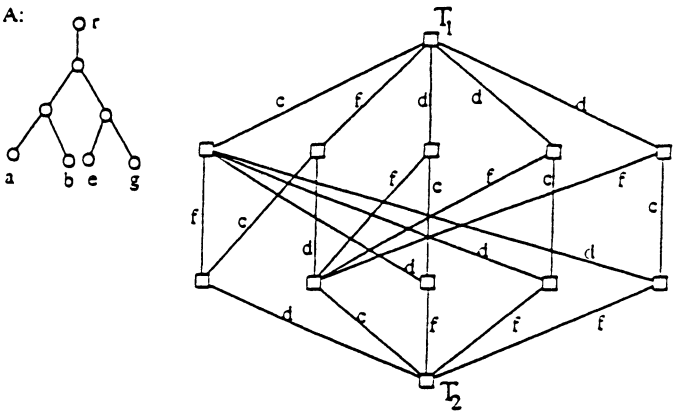


Figure 4
(T_1, T_2) interval for a given agreement subtree A

4. The modified metric and intervals.

Although the metric d in the previous section is easy to compute because of the existence of polynomial algorithms, it does not take into account the location of pruned leaves. In [4] d was modified by adding a fractional part, which reflects how far apart the pruned leaves are, with respect to an optimal agreement subtree.

We next recall some material from [3]. Suppose T_1 and T_2 belong to $\mathcal{T}_n$ and A is an agreement subtree for T_1 and T_2. Let P_A denote the set of all leaves pruned from T_1 and T_2 to obtain A. For each $x \in P_A$, let $A_i(x)$ denote the tree obtained from T_i by pruning all leaves in P_A except x (i = 1, 2). Now let $A(x)$ be the tree formed by adding leaves x_1 and x_2 to A such that $A(x)$ is isomorphic to $A_1(x)$ after pruning x_2 and is isomorphic to $A_2(x)$ after pruning x_1. Notice that $A(x)$ has $n + 1$ leaves and therefore belongs to $\mathcal{T}_{n+1}$. See Figure 5 for an illustration.

Let $s(x, A)$ denote the length of the path between the leaves x_1 and x_2 in $A(x)$. In the example, $s(x, A) = 4$. Define

$$\lambda(A) = \sum_{x \in P_A} s(x,A), \tag{9}$$

and

$$L(T_1, T_2) = \min_{A \in \mathcal{A}(T_1,T_2)} \lambda(A). \tag{10}$$

Finally we set

$$d_1(T_1, T_2) = d(T_1, T_2) + \frac{1}{n\, d(T_1,T_2)} L(T_1, T_2) \text{ if } T_1 \neq T_2, \text{ and} \tag{11}$$

$$d_1(T_1, T_2) = 0 \text{ if } T_1 = T_2$$

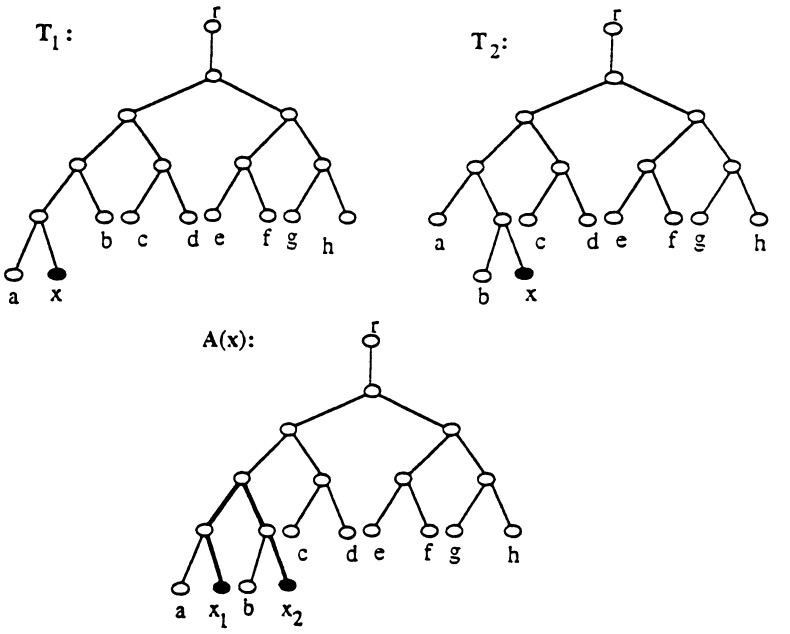


Figure 5

The function d_1: $\mathcal{T}_n \times \mathcal{T}_n \to \mathbf{R}$ is indeed a metric as was shown in [3]. Also a dynamic programming approach is used to produce an agreement subtree A which minimizes $\lambda(A)$ in polynomial time. Such an agreement subtree A is said to be *optimal*. Since an optimal agreement subtree is efficiently computable, then $d_1(T_1, T_2)$ can be computed in polynomial time for any T_1 and T_2 in $\mathcal{T}_n$.

Define the interval between T_1 and T_2 as before:

$$\mathcal{I}_1(T_1, T_2) = \{T \in \mathcal{T}_n : d_1(T_1, T) + d_1(T, T_2) = d_1(T_1, T_2)\}. \tag{12}$$

The characterization of the intervals for the metric d_1 is much easier than for the metric d. In the proof of the triangle inequality for d_1, presented in [3], it is easy to observe that the strict inequality $d_1(T_1, T_2) < d_1(T_1, T) + d_1(T, T_2)$ holds unless $T \equiv T_1$ or $T \equiv T_2$. Therefore we have the following result.

Theorem 2. *For any T_1 and T_2 in $\mathcal{T}_n$, $\mathcal{I}(T_1, T_2) = \{T_1, T_2\}$.*

Some researchers might feel that having only trivial intervals is a strange property for a metric. Later, a new metric d_2 will be defined which, despite its similarity to d_1, will have non-trivial intervals. In the definition of the metric d_1 (and also d_2) the function L is present. Considering how L is defined, this function can be regarded as a measure of dissimilarity of two trees T_1 and T_2. Although L is reflexive and symmetric, it unfortunately does not satisfy the triangle inequality and therefore is not a metric.

In Figure 6 we present an example of three trees $T_1, T_2, T_3 \in \mathcal{T}_{52}$ for which the triangle inequality is not satisfied. In this example, U is the full binary tree of 16 leaves. The labels of those leaves are identical in all three trees T_1, T_2, and T_3, and a similar assumption is made for V and W.

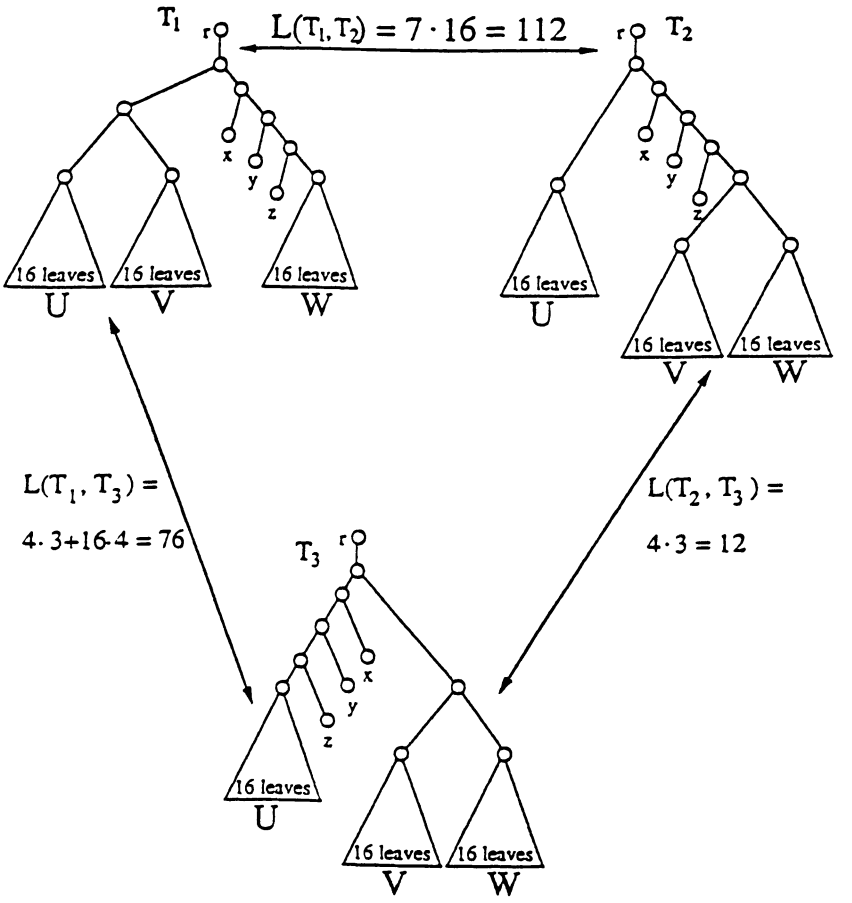


Figure 6

Let T_1 and T_2 be two trees from $\mathcal{T}_n$. Using the same notation as in the definition of d_1, we define

$$d_2(T_1, T_2) = d(T_1, T_2) + \frac{1}{n^2} L(T_1, T_2). \tag{13}$$

Comparing d_2 to the metric d_1, the function d_2 simply has a different normalization factor in front of $L(T_1, T_2)$. Of course, because n_2 is a strict upper bound for L, the second term is a fraction less than 1. Figure 7 illustrates these differences.

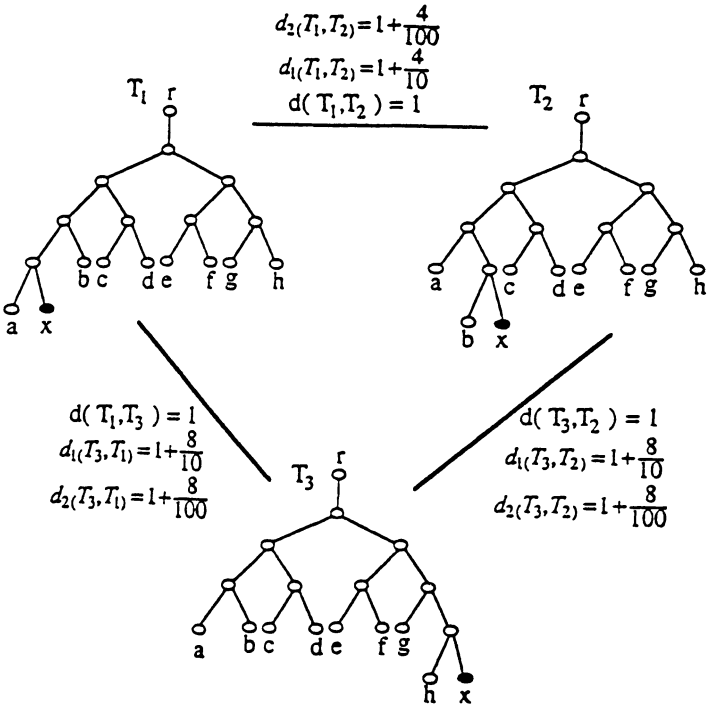


Figure 7

Theorem 3. *The function* d_2: $\mathcal{T}_n \times \mathcal{T}_n \to \mathbf{R}$ *is a metric on* $\mathcal{T}_n$.

PROOF. Obviously, the function d_2 is nonnegative and symmetric so that we need only show the triangle inequality. For any T_1 and T_2, we have $d_2(T_1, T_2) < d(T_1, T_2) + 1$. Therefore, if $d(T_1, T_3) + d(T_3, T_2) > d(T_1, T_2)$, then $d(T_1, T_3) + d(T_3, T_2) > d_2(T_1, T_2)$, and also $d_2(T_1, T_3) + d_2(T_3, T_2) > d_2(T_1, T_2)$.

Now assume that $d(T_1, T_3) + d(T_3, T_2) = d(T_1, T_2)$. Let A_{ij} be an agreement subtree for the trees T_i and T_j realizing the minimum in the definition of $d_2(T_i, T_j)$. Let $v_1, v_2, \ldots, v_m$ be the leaves to be pruned from T_1 (and T_3) to obtain A_{13} and let $u_1, u_2, \ldots, u_k$ be the leaves to be pruned from T_2 (and T_3) to obtain A_{23}. Since $d(T_1, T_3) + d(T_3, T_2) = d(T_1, T_2)$ the sets $\{v_1, v_2, \ldots, v_m\}$ and $\{u_1, u_2, \ldots, u_k\}$ are disjoint. If A denotes the tree obtained by pruning the leaves $v_1, v_2, \ldots, v_m, u_1, u_2, \ldots, u_k$ from both T_1 and T_2, then notice that A can be also obtained from either A_{13} or A_{23} by pruning. Therefore, $s(v_i, A) \leq s(v_i, A_{13})$ for $i = 1, 2, \ldots, k$ and $s(u_j, A) \leq S(u_j, A_{23})$ for $j = 1, 2, \ldots, k$. Finally, we obtain the triangle inequality from the following:

$$
\begin{aligned}
d_2(T_1, T_2) &\le d(T_1, T_2) + \frac{1}{n^2}\left[\sum_{i=1}^{m} s(v_i, A) + \sum_{j=1}^{k} s(u_j, A)\right] \\
&= d(T_1, T_3) + \frac{1}{n^2}\sum_{i=1}^{m} s(v_i, A) + d(T_3, T_2) + \sum_{j=1}^{k} s(u_j, A) \qquad (14) \\
&\le \left[d(T_1, T_3) + \frac{1}{n^2}\sum_{i=1}^{m} s(v_i, A_{13})\right] + \left[d(T_3, T_2) + \sum_{j=1}^{k} s(u_j, A_{23})\right] \\
&= d_2(T_1, T_3) + d_2(T_3, T_2) \quad \blacksquare
\end{aligned}
$$

The next theorem characterizes the interval $\mathcal{I}_2(T_1, T_2)$ for the metric d_2.

Theorem 4. *Let T_1 and T_2 be arbitrary trees from $\mathcal{T}_n$. A tree T lies on the interval $\mathcal{I}_2(T_1, T_2)$ with respect to d_3 if and only if there is an optimal agreement subtree A of T_1 and T_2 and a bipartition U, V of the pruned vertices such that $T \in U \oplus_{T_1} A \oplus_{T_2} V$ and for every $u \in U$ and $v \in V$ the paths u_1u_2 in $A(u)$ and v_1v_2 in $A(v)$ have no common internal vertices of A.*

PROOF. Suppose first that there is an optimal agreement subtree A of T_1 and T_2 and a bipartition U, V of pruned vertices having the required properties. Let T belong to $U \oplus_{T_1} A \oplus_{T_2} V$. Notice that $A_{13} \equiv U \oplus_{T_1} A$ is an agreement subtree of T_1 and T, and $A_{23} \equiv A \oplus_{T_2} V$ is an agreement subtree of T_2 and T. Because the paths u_1u_2 in $A(u)$ and v_1v_2 in $A(v)$ are disjoint for every $u \in U$ and $v \in V$, we have that $s(v, A) = s(v, A_{13})$ for every vertex $v \in V$. Similarly, $s(u, A) = s(u, A_{23})$ for every $u \in U$. Therefore, the two weak inequalities occuring in (14) in the last part of the proof of Theorem 3 become the equalities (T plays the role of T_3) and $d_2(T_1, T) + d_2(T, T_2) = d_2(T_1, T_2)$. This means that $T \in \mathcal{I}_2(T_1, T_2)$.

For the converse, suppose that T_1, T_2, and $T = T_3$ satisfy $d_2(T_1, T) + d_2(T, T_2) = d_2(T_1, T_2)$. Then in (14) both weak inequalities become the equalities. The first one implies that A is an optimal agreement subtree of T_1 and T_2. Because $d_2(T_1, T) + d_2(T, T_2) = d_2(T_1, T_2)$ implies the corresponding equality for d, we have that $T \in \mathcal{I}(T_1, T_2)$ with respect to the metric d. From Theorem 1, there exist a bipartition U, V of the pruned vertices such that $T \in U \oplus_{T_1} A \oplus_{T_2} V$. Of course, $A_{12} \equiv A \oplus_{T_1} U$ is an agreement subtree of T_1 and T, $A_{23} \equiv A \oplus_{T_2} V$ is an agreement subtree of T_2 and T. The second equality in (14) implies that $s(v, A) = s(v, A_{13})$ for all $v \in V$ and $s(u, A) = s(u, A_{12})$ for all $u \in U$. This forces the condition that the u-path in $A(u)$ and the v-path in $A(v)$ have no common internal vertices. $\blacksquare$

In Figure 8 the interval $\mathcal{I}_2(T_1, T_2)$ is presented for some trees T_1 and T_2 as in Figure 3. Notice that this interval is much smaller than the composing interval with respect to the metric d.

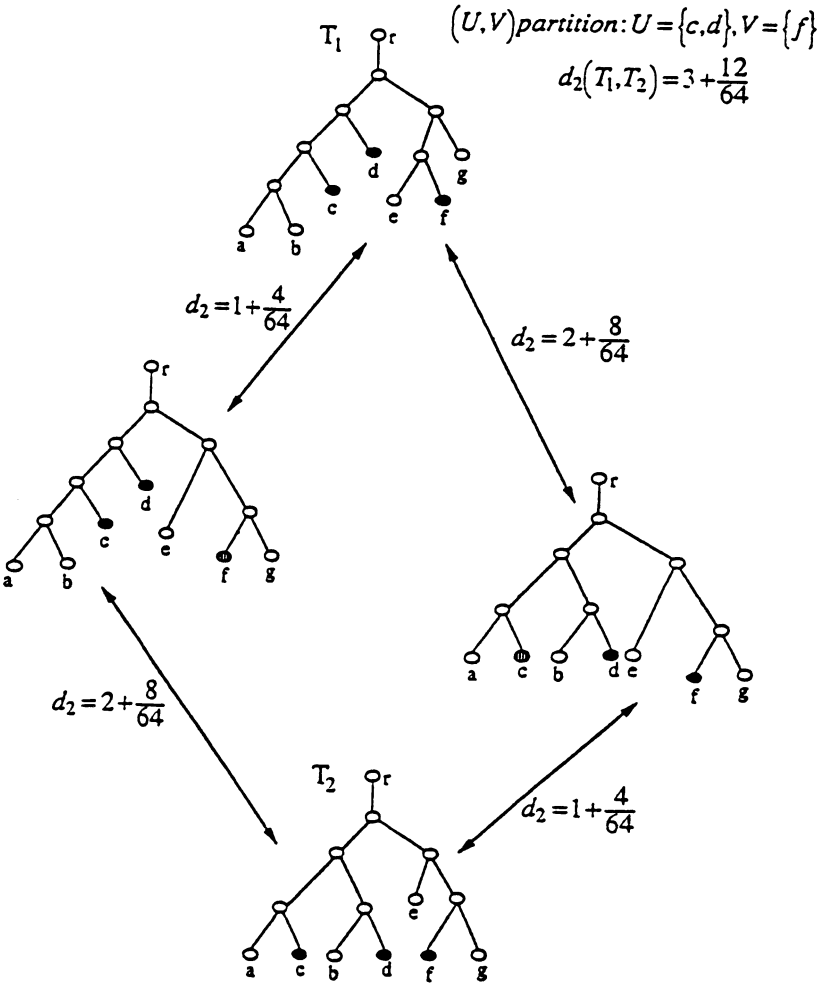


Figure 8

6. References

1. A. Amir and D. Keselman, *Maximum Agreement Subtree in a Set of Evolutionary Trees*, preprint, College of Computing, Georgia Institute of Technology, 1994.
2. M. Farach, T. M. Przytycka, and M. Thorup, On the agreement of many trees, *Inf. Proc. Lett.* 55 (1995), 297-301.
3. W. D. Goddard, E. Kubicka, G. Kubicki, and F. R. McMorris, The agreement metric for labeled binary trees, *Math. Biosci.* 123 (1994), 215-226.
4. W. D. Goddard, E. Kubicka, G. Kubicki, and F. R. McMorris, Agreement subtrees, metric and consensus for labeled binary trees, in *Partitioning Data Sets*, I. Cox, P. Hansen, B. Julez, Eds., DIMACS Series, Vol. 19, AMS, Providence, RI, 1995, 97-104.
5. M. Steel and T. Warnow, Kaikoura tree theorems: computing the maximum agreement subtree, *Inf. Proc. Lett.* 48 (1993), 77-82.

Department of Mathematics, University of Louisville, Louisville, KY 40292
E-mail address: emkubi01@homer.louisville.edu, gmkubi01@homer.louisville.edu, frmcmo01@homer.louisville.edu

https://doi.org/10.1090/dimacs/037/17

DIMACS Series in Discrete Mathematics
and Theoretical Computer Science
Volume **37**, 1997

Sparse Dynamic Programming for Maximum Agreement Subtree Problem

Teresa M. Przytycka

ABSTRACT. The Maximum Agreement Subtree (MAST) arises in biology as a measure of consistency between two evolutionary trees . As many optimization problems, the problem of computing MAST can be solved using dynamic programming approach. A straightforward dynamic programming algorithm for the MAST problem takes $O(n^2)$ time. The complexity can be reduced using the observation that the dynamic programming instances arising in the MAST problem are sparse. This property of has been explored to obtain several subquadratic time algorithms. In this paper we first review the key sparsification techniques used in subquadratic time *MAST* algorithms. Subsequently we develop a new $O(n \log n\sqrt{d})$ time algorithm for trees with degree bounded by d. For $d < \log n$ the algorithm takes $O(n \log n \log d + nd\sqrt{d})$ time. This improves upon previous complexity bounds for this case and fills the gap between the $O(n \log n)$ time algorithm for binary trees and the $O(n\sqrt{n} \log n)$ time algorithm for unbounded degree trees.

1. Introduction

The Maximum Agreement Subtree arises in biology as a measure of consistency between two evolutionary trees. An *evolutionary tree* for a species set L is a rooted tree whose leaves are uniquely labeled by the species in L, and (unlabeled) internal nodes represent ancestor species. We refer to such a tree as a *leaf labeled rooted tree.* The set of leaf labels of a tree T is denoted $L(T)$. The *size*, n, of a tree is defined to be the number of leaves.

For a set L' such that $L' \subseteq L(T)$,we define the *topological restriction of* T *to* L', written $T|L'$, to be the tree obtained form T by removing leaves not in L' and contracting degree one nodes. Formally, the nodes of $T|L'$ are defined by pairs $\{\mathtt{lca}^T(a,b) | (a,b) \in L' \times L'\}$ and the arcs are defined in such a way that for all $(a,b) \in L' \times L'$, $\mathtt{lca}^{T|L'}(a,b) = \mathtt{lca}^T(a,b)$. Here $\mathtt{lca}^T(x,y)$ denotes the least common ancestor of nodes x and y in T. The tree $T|L'$ is uniquely determined by

---

1991 *Mathematics Subject Classification.* 05C35.

Supported partially by Sloan and Department of Energy Postdoctoral Fellowship for Computational Biology and by grant GM29458. A part of this research was done while visiting Dept. of Computer Sciences, University of Maryland, College Park. The author thanks Martin Farach and Mikkel Thorup for helpful discussions.

the tree T and the set L'. Given a tree T and a set $L' \subseteq L(T)$ the tree $T|L'$ can be computed in $O(n)$ time [**5**].

Given two leaf labeled trees T_0, T_1 with the same set of leaf labels L, the *Maximum Agreement Subtree* problem is to compute a maximum cardinality subset L' of L such that $T_0|L'$ and $T_1|L'$ are isomorphic. (Isomorphism of leaf labeled trees is assumed to preserve the labels.) Maximum Agreement Subtree problem extends in a natural way to the case of more than two trees and to unrooted trees.

The first MAST algorithm was given by Finden and Gordon [**8**]. Their algorithm is a heuristic algorithm that takes $O(n^5)$ time and does not guarantee an optimal solution. Subsequently, Kubicka *et al.* [**12**] presented an $O(n^{(\frac{1}{2}+\epsilon)\log n})$ time algorithm for the binary *MAST* problem. The first polynomial time algorithms for the *MAST* problem were given by Steel and Warnow [**15**] and independently by Goddard, Kubicka and McMorris [**13**]. The Steel-Warnow algorithm uses dynamic programming technique and runs in $O(n^2)$ time on bounded degree rooted or unrooted trees and in $O(n^4 \log n)$ time on unbounded degree unrooted trees. Subsequently, Farach and Thorup gave an $O(n^2 c^{\sqrt{\log n}})$ time algorithm for the *UMAST* problem [**5**] and an $O(n^{1.5} \log n)$ time algorithm for the rooted version of the problem [**6**]. The algorithm of [**6**] provides an $O(n^{1+\epsilon})$ time reduction of the *MAST* problem to the Unary Weighted Bipartite Matching (UM) problem [**9**]. At the same time they show that the *MAST* problem is at least as hard as the UM problem.

In practice evolutionary trees have small degrees, and most frequently they are simply binary trees, thus the *MAST* problem for binary and bounded degree trees are of particular interest. Farach, Przytycka and Thorup [**3**] gave an $O(n \log^3 n)$ time algorithm for the Maximum Agreement Subtree problem for binary trees. Subsequently, Kao [**11**] gave an $O(n \log^2 n)$ algorithm for the same problem. Kao's algorithm takes $O(\min\{nd^2 \log d \log^2 n, nd^{\frac{3}{2}} \log^3 n\})$ for degree d trees. Finally Cole and Hairhanan [**2**] improved the algorithm of Farach, Przytycka and Thorup to an $O(n \log n)$ time algorithm (see also [**1**]).

In this paper we discuss sparse dynamic programming algorithms for computing *MAST* for a pair of rooted trees. We survey various sparsification results using one general framework. This allows us to simplify some of the earlier arguments. Reversing the historical development of the algorithms for the *MAST* problem, we start with an outline the $O(n \log n)$ time algorithm of Cole, Farach, Hairhanan, Przytycka and Thorup[**1**]. Subsequently we generalize this algorithm to non-binary trees. The fact that the algorithm can be generalized to non-binary trees was suggested in [**3, 1, 16**]. In [**1**] we concluded that such a generalization to degree d trees appears to yield an algorithm with running time $O(\min\{n\sqrt{d} \log^2 n, nd \log n \log d\})$. In the same paper an algorithm with running time $O(n\sqrt{d} \log n)$ was conjectured. In this paper, we show that this conjecture is true. For the case where $d < \log n$, our algorithm takes $O(n \log n \log d + nd\sqrt{d})$ time.

1.1. Sparse dynamic programming. The top level idea behind the sparse dynamic programming approach can be described as follows. Let T_1, T_2 be two binary trees. We say that a pair (x_1, x_2), where x_i is a node of tree T_i *dominates* a pair (x_1', x_2') where x_i' is a node of tree T_i if x_i is an ancestor of x_i'. Furthermore, if $x_i \neq x_i'$ for $i = 1, 2$ then we say that (x_1, x_2) *properly dominates* (x_1', x_2').

For a pair of internal nodes x_1, x_2 where $x_i \in T_i$, define $MAST(x_1, x_2)$ to be a maximum agreement subtree of the subtrees rooted at x_1 and x_2 respectively. The quadratic dynamic programming algorithms for the $MAST$ problem relay on the fact, that given a pair of nodes (x_1, x_2) where none of x_i is a leaf, $MAST(x_1, x_2)$ can be computed in a constant time from the $MAST$ values for a constant number of pairs dominated by (x_1, x_2) (see next section for the precise algorithm). Using this observation the quadratic dynamic programming algorithm proceeds by computing all entries (i, j) of an $(2n-1) \times (2n-1)$ array, $MAST$ where $MAST(i, j) = MAST(v_i, u_j)$. The entries are computed using any partial order which agrees with the dominance partial order defined above.

Subquadratic time algorithms for the $MAST$ problem are based on the observation that most of the entries of the array $MAST$ are not relevant for computing the target value $MAST(u_{2n-1}, v_{2n-1})$. In fact, the matrix of the relevant values is sparse. We refer to a dynamic programming algorithm that produces such a sparse table of values as to a *sparse dynamic programming* algorithm.

In a sparse dynamic programming algorithm, we can no longer assume that each entry of the $MAST$ array can be computed in constant time from its other entries. Neither we can assume that entries of the $MAST$ values can be located in constant time. Thus, two steps will be needed. First, we will explore the structure of the problem to decide which entries of the dynamic programming array are needed. Subsequently, we address the problem of storing and accessing the relevant data.

2. Reduction of the number of entries

Consider the following $O(n^2)$ time dynamic programming algorithm for computing $MAST$ value of two trees T_1, T_2 rooted at R_1, R_2 respectively [**15**]. The algorithm processes all pairs of vertices (x_1, x_2), where $x_i \in T_i$, in any order that agrees with the dominance order. For an internal node x we use $side(x)$ and $cntr(x)$ to denote the two children of x. For the purpose of the later algorithms, we let the *central child*, denoted $cntr(v)$, be the node with the maximum number of descending leaves, and we let the *side child*, denoted $side(v)$ be the other child, with ties broken arbitrarily. Correspondingly, call the arc $(x, cntr(x))$ a *central arc*.

The algorithm uses the following dynamic programming formula:

$$
(2.1)\quad MAST(x_0, x_1) = \max \begin{cases} binvalue(L(x_0) = L(x_1)) \\ \quad \text{if } x_0 \text{ and } x_1 \text{ are leaves.} \\ MAST(x_0, cntr(x_1)) \\ \quad \text{if } x_1 \text{ is not a leaf.} \\ MAST(cntr(x_0), x_1) \\ \quad \text{if } x_0 \text{ is not a leaf.} \\ MAST(x_0, side(x_1)) \\ \quad \text{if } x_0 \text{ is not a leaf.} \\ MAST(side(x_0), x_1) \text{ if } x_1 \text{ is not a leaf.} \\ MAST(cntr(x_0), cntr(x_1)) + \\ MAST(side(x_0), side(x_1)) \\ \quad \text{if neither } x_0 \text{ nor } x_1 \text{ is a leaf.} \\ MAST(side(x_0), cntr(x_1)) + \\ MAST(cntr(x_0), side(x_1)) \\ \quad \text{if neither } x_0 \text{ nor } x_1 \text{ is a leaf.} \end{cases}
$$

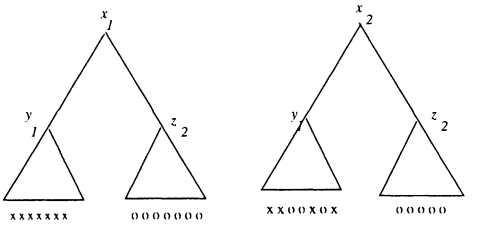


FIGURE 1. A non-interesting entry (x_1, x_2).

We say that two subtrees S_1, S_2 of T_1, T_2 respectively *intersect* if $L(S_1) \cap L(S_2) \neq \emptyset$. To see that some entries of the array $MAST$ are not relevant in computing $MAST$ value, consider the example illustrated in Figure 1. Here subtrees rooted at y_1 and z_2 do not intersect. In this case for any node z' such that z_2 is an ancestor of z we have $MAST(y_1, z) = 0$. Thus we do not need to compute the $MAST$ value for a pair (x_1, x_2) unless there exist a pair of leaves (a, b) such that $\mathtt{lca}^{T_i}(a, b) = x_i$.

However, after this simple reduction of the number of entices described above we may still be left with $\Omega(n^2)$ entries (see Figure 2 a).

Let $a, b, c \in L$. We say that a triple a, b, c is *resolved* if and only if $T_1|\{a, b, c\} = T_2|\{a, b, c\}$ and it has two internal nodes. Intuitively, each agreement subtree (of size at least 3) represents a set of consistent, resolved triples. This suggest another natural candidate for an interesting entry: an entry that corresponds to a pair of roots of a resolved triple. Note that even if we restrict the set of interesting entries to the roots of resolved triples the number of such pairs remains, in the worst case, quadratic. (See Figure 2 b).

To reduce the number of interesting entries the subquadratic time algorithms **[1, 2, 3, 6, 11]** organize computation along pairs of paths (P_1, P_2) where P_i is a path in T_i.

By a *centroid path* we mean a maximum path using central arcs. The vertex of a centroid path, P, that is closest to the root is called the *topmost* vertex of this path and is denoted by $topmost(P)$.

A centroid path P of a tree T dominates a centroid path P' of the same tree if $topmost(P)$ is an ancestor of $topmost(P')$. The *depth* of a centroid path is the number of centroid paths properly dominating it. Note that the depth of any centroid path is at most $\log n$.

We have a natural partial order on pairs of centroid paths: a pair (P_1, P_2) of centroid paths *dominates* a pair (P_1', P_2') of centroid paths where P_i, P_i' is a centroid path of T_i, if and only if $(topmost(P_1), topmost(P_2))$ dominates $(topmost(P_1'), topmost(P_2'))$.

Note that each vertex belongs to a unique centroid path. Thus, if we consider all pairs of centroid paths then every pair of nodes belongs to exactly one such pair of paths.

A pair of centroid paths (P_1, P_2) is an *interesting pair of paths* if there exists a pair of labels a, b such that $\mathtt{lca}^{T_1}(a, b) \in P_1$ and $\mathtt{lca}^{T_2}(a, b) \in P_2$.

A pair of internal nodes (x_1, x_2) where $x_i \in T_i$ is *interesting* if it belongs to an interesting pair of paths and the side trees $side(x_1)$ and $side(x_2)$ intersect. A pair (x_1, x_2) where one of x_i, say x_2, is a leaf is interesting if $L(x_2) \in L(x_1)$.

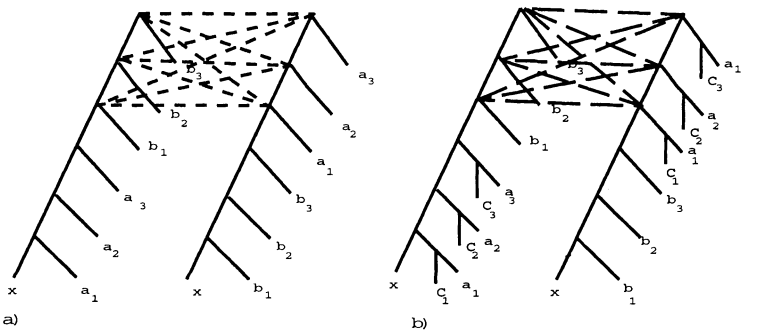


FIGURE 2. An example showing that a simple reduction of entries does not suffice.

LEMMA 2.1. *Let T_1, T_2 be two trees such that the maximum depth of a centroid path in any of the trees is k. Then there are $O(kn)$ interesting pairs of nodes in (T_1, T_2).*

PROOF. Divide the interesting pairs into three groups: A *left non-singleton interesting pair* is an interesting pair (x_1, x_2) such that there exist an interesting pair (x_1, x_2') with $x_2' \neq x_2$ and x_2' belonging to the same centroid path as x_2. A *right non-singleton interesting pair* is defined in a symmetric way. The remaining pairs are called *singleton pairs.* First, we will show that the number of left non-singleton pairs is $O(kn)$. The argument for right non-singleton interesting pairs is symmetrical. Let $P_1 = u_1, \ldots u_p$ be the centroid path of T_1 starting at the root of T_1. Let $m_i = |L(side(u_i))|$. For any u_i there are $O(m_i)$ left non-singleton interesting pairs of the form (u_i, x_j). Since $\sum_i m_i \leq n$ and the number of levels is at most k, the bound for non-singleton pairs follows. Consider an interesting singleton pair (u_i, v_i). Let u_i belong to P_1 and v_i belong to P_2. By the definition of an interesting pair of paths, there exist another interesting pair (u_j, v_j) where $u_i \neq u_j$ and $v_i \neq v_j$ that belongs to the same pair of paths. Thus if a pair (u_i, v_j) where u_i belongs to P_1 and v_j is ancestor of v_i is interesting, then it must be a non-singleton interesting pair. Thus the number of singleton pairs is also $O(kn)$. □

COROLLARY 2.2. In a pair of leaf labeled trees of size n, there are $O(n \log n)$ interesting pairs of nodes.

LEMMA 2.3. *All interesting pairs of nodes for a pair of trees T_1, T_2 can be identified in $O(n \log n)$ time.*

PROOF. First, identify the topmost centroid path $P_1 = u_1, \ldots u_p$ of T_1. For any side subtree $side(u_i)$ find recursively all interesting pairs in the pair of subtrees $(side(u_i), T_2 | L(side(x_0))$. Then it remains to find interesting pairs where the first element of the pair belongs to P_1. Assume that in the recursive step we stored one interesting pair per each intersecting pair of side trees $(side(u_i), side(x))$ where u_i belongs to P_1 and x belongs to T_2. Then reporting of all interesting pairs of type (u_i, x) reduces to traversing the tree T_2 in postorder. For each centroid path we check if it forms an interesting pair of paths with P_1 and if so report all intersecting pairs of subtree as interesting pairs. □

In our algorithm, we compute in addition to *MAST* values for interesting pair the *MAST* values for so called *supporting pairs.* If (x_1, x_2) is an interesting pair,

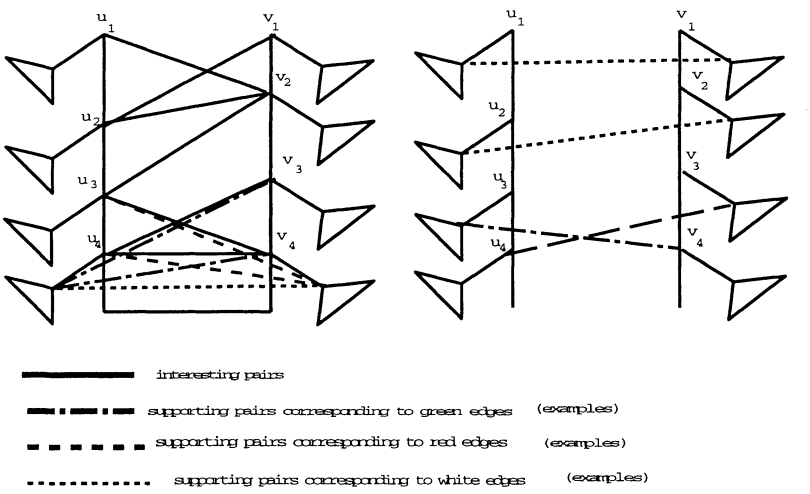


FIGURE 3. The relation of the path matching and an agreement mapping

then we call pair $(x_1, topmost(x_2))$ a *red supporting pair*, $(topmost(x_1), x_2))$ a *green supporting pair*, and the pair $(topmost(x_1), topmost(x_1))$ a *white supporting pair.* Clearly, the number of supporting pairs is at most three times the number of interesting pairs.

3. Sparse dynamic computation of *MAST* value for binary trees

Let $L' \subseteq L(T)$. If $T_1|L'$ is isomorphic to $T_2|'_L$ then $T_1|L'$ is called an *agreement subtree* and the isomorphism between T_1 and T_2 is called an *agreement mapping.* We say that a subtree S_1 of T_1 is mapped onto subtree S_2 of T_2 in this agreement mapping if $S_1|L' \cap L(S_I)$ is nonempty and is isomorphic to $S_2|L' \cap L(S_2)$.

Consider a pair P_1, P_2 of centroid paths of T_1 and T_2 respectively and consider a possible agreement mapping of side subtrees (see Figure 3). Clearly if $v_{i_1}, \ldots v_{i_k}$ and are nodes of P_1 that are mapped to nodes $u_{i_1}, \ldots u_{i_k}$ of P_2 in the agreement mapping then for $j < k$ $side(v_i)$ is mapped into $side(u_i)$. Thus, in particular, for $i < k$ the pairs (v_{i_j}, u_{i_j}) have to be interesting. For the last pair (v_{i_k}, u_{i_k}) (the pair (u_4, v_4) in Figure 3) it is also possible that the side tree of one node is mapped onto central subtree of the other node and vice versa. In this case the pair (v_{i_k}, u_{i_k}) does not need be interesting according to our definition. (This is the case for the pair (v_3, u_3) in Figure 3.) However, our algorithm needs to know the value of $MAST(v_3, u_3)$. We will show how to store enough information to compute such a value in an efficient way. The process of computing a best agreement matching for an interesting pair of centroid path is called the *path matching.*

Let $P_1 = u_1, \ldots u_p, P_2 = v_1, \ldots v_q$ be an interesting pair of centroid paths. Assume that the *MAST* value for any supporting dominated properly by an interesting pair (x_1, x_2) where $x_i \in P_i$ is known. The problem of computing *MAST* values for all interesting and supporting pairs of a given pair of interesting paths can be formalized as a certain type of a bipartite matching problem, called the *path matching problem* [**1**].

Let $G(P_1, P_2) = (L \cup R, E)$ be a bipartite multigraph, where each multiedge consists of three edges: a *white* edge, a *red* edge and a *green* edge. Each of these edges has a weight associated with it. For any interesting pair (u_i, v_j) there is a multiedge between vertices corresponding to u_i and v_j. The white edge in this multiedge has weight equal to $MAST(side(u_i), side(v_j))$, the red edge has weight equal to $MAST(side(u_i), cntr(v_i))$ and the green edge has weight equal to $MAST(cntr(v_i), side(v_j))$. Furthermore there is a multiedge between u_p and any v_j if $u_p \in L(v_j)$. Finally, there is a multiedge between u_p and v_q if both these leaves have the same label. The weights of all edges in these multiedges are one. The edges adjacent to u_q are defined in a symmetric way.

Observe the following relation between edges of the graph $G(P_1, P_2)$ and supporting pairs dominated by the interesting pairs of (P_1, P_2). Consider for example a white edge (u_i, v_j). Let P_1' and P_2' be the centroid paths of $side(u_i)|side(v_j)$ and $side(v_j)|side(u_i)$ respectively. (Recall that centroid path in a subtree is inherited form centroid path of the tree). Then the weight of a white edge is equal to the $MAST$ value for the white supporting pair in this pair of paths. Similarly the weight of a green edge is equal to the $MAST$ value of a corresponding green supporting pair etc. (compare Figure 3). Similarly each read and green edges corresponds to supporting pair of the same color.

We define a *proper crossing* in $G(P_1, P_2)$ to be a red-green edge pair in G such that the two edges cross and the endpoint of the green edge in L is above that of the red edge.

An Agreement Matching in $G(P_1, P_2)$. A matching in $G(x)$ is an agreement matching if:

1. It has zero or more white edges and at most one proper crossing.
2. No white edge crosses any other edge; further, all white edges properly dominate the edges in the proper crossing, if any.

The weight of an agreement matching is the sum of the weights of its edges.

Each such agreement matching defines in a unique way mapping between nodes of P_1 and P_2 in an agreement tree. Namely, the endpoints of white edges are mapped one to another. Furthermore, if the mapping contains a proper red-green crossing then the left end point of the green edge is mapped into right endpoint of the red edge (see Figure 3 b). More precisely, recall that each white, green or red edge corresponds to a unique supporting pair. Choosing a given edge in the agreement mapping corresponds to mapping the subtrees rooted at the elements of the corresponding supporting pair one to another. A maximum weight agreement matching corresponds to a maximum weight agreement subtree.

The $MAST$ algorithm of [**1**] processes interesting pairs of paths in an order that agrees with the dominance order. For each such pair (P_1, P_2) of interesting paths it computes a maximum weight agreement matching for the graph $G(P_1, P_2)$. Since we need to know the weight of the edges of $G(P_1, P_2)$ prior to starting the path matching algorithm, we modify slightly graph $G(P_1, P_2)$ by adding edges corresponding to supporting pairs that belong to (P_1, P_2). The number of edges in $G(P_1, P_2)$ increases at most three times. Since the maximum agreement matching algorithm computes for every edge e in G a maximum agreement matching restricted to the set of edges dominated by e the $MAST$ values for these supporting pairs will be computed as a partial result of the path matching algorithm. Thus processing the pairs of interesting paths in an order that agrees with the dominance

order ensures that prior to starting the path matching algorithm for a given pair of paths, the weights of all edges in the corresponding graph G are computed.

LEMMA 3.1. [**3**] *Given two centroid paths the path matching problem can be solved in $O(m \log n)$ time where m is the number of edges in the bipartite graph.*

PROOF. **(Outline.)** The approach is similar to the $O(n \log n)$ time algorithm for the heaviest common subsequence [**10**] with the necessary complication introduced by allowing a proper red-green crossing. (Without that crossing allowed the problem would be equivalent to the heaviest common subsequence problem).

Construct balanced binary search tree S whose leaves are the vertices in R. Next, the vertices in L are considered in turn in bottom-to-top order. For each vertex $u_i \in L$, the vertices adjacent to it in R are searched for in S and the edges incident with u_i are considered in the bottom-up order with respect to their left endpoint.

For each white edge the largest weight agreement matching for the set of edges dominated by it is computed. The following information is stored at each vertex z of S. Let $anc(z)$ denote the set of ancestors of z in S, z inclusive and $S(z)$ is the subtree of T rooted at z.

1. $g(z)$: For each z, $\max_{z' \in anc(z)} g(z')$ will be the heaviest green edge in S which forms a proper crossing with each red edge in $S(z)$.
2. $x(z)$: This is the largest weight proper crossing among the edges in $S(z)$.
3. $m(z)$: This is the largest weight agreement matching containing a white edge such that the topmost white edge is in $S(z)$.
4. $y(z)$: This is the largest weight proper crossing such that the green edge in this crossing is in S but not in $S(z)$, the red edge in this crossing is in $S(z)$, and the green edge does not form a proper crossing with all the red edges in $S(z)$.
5. $r(z)$: This is the heaviest red edge in $S(z)$.

Let e be a processed edge. The information stored so far in S takes into account only the edges processed before edge e. From this information, the largest weight agreement matching containing only edges dominated by e is computed. The computation time is proportional to the depth of the right endpoint of e in S and thus takes $O(\log n)$ time. In the same time the information stored in S is updated so that it takes into account the edge e. The details can be found in [**1**]. □

Outline of the $O(n \log n)$ time algorithm for computing $MAST$ of two binary trees. Let BAgree be algorithm that computes $MAST$ values for all interesting and supporting pairs for a pair of binary trees. The algorithm consists of three parts:

Algorithm BAgree

1: Let $P_1 = u_1, u_2, \ldots u_p$. Compute $S_i = T_2|L(side(u_i))$. This can be done in linear time [**6**].
2: For each pair of trees $(side(u_i), S_i)$ compute recursively BAgree$(side(u_i), S_i)$.
3: Solve the Path Matching problem for every interesting pair of paths P_1, P_2' where P_2' is a centroid path of T_2 in an order agrees with the dominance partial order.

THEOREM 3.2. [**2, 1**] *The MAST for two binary trees can be computed in* $O(n \log n)$ *time.*

PROOF. **(Outline)** Note that a direct application of Lemmas 3.1 and 2.1 leads to an $O(n \log^2 n)$ time algorithm. To achieve an $O(n \log n)$ time algorithm Cole and Hairhanan replaced binary search tree used in the path matching algorithm by a weighted binary search tree. The weights are chosen in such a way that amortized complexity of all path matching problems is $O(n \log n)$. For the details of the analysis we refer the reader to [**1**]. □

In the next section, we will show, how to generalize this algorithm to higher degree trees. For this purpose we need to extract some properties of the path matching algorithm:

LEMMA 3.3. *For each edge e processed by the path matching algorithm the following queries can be answered at the time of processing e without increasing the approximate cost of the algorithm.*

- *the weight of the maximum weight proper crossing dominated by e*
- *maximum weight path matching value for the edges properly dominated by e.*
- *for each green edge a maximum weight green edge which has the same left endpoint as e and which is dominated (not necessarily properly) by e.*

PROOF. All these properties are immediate consequences of the path matching algorithms [**1, 2, 3**]. The first two are the base of the dynamic programming algorithm used in the algorithm. The last one follows directly from the order of processing the edges. □

4. Non-binary trees

In the case of non-binary trees, the equation (2.1) generalizes as follows. For a pair of internal nodes (x_1, x_2) consider bipartite graph $B(x_1, x_2) = (B_1, B_2, E)$ where each vertex of B_i corresponds to the subtree of T_i rooted at a child of x_i. There is an edge between b_1 and b_2 where $b_i \in B_i$ if the subtrees rooted at b_1 and b_2 intersect. The weight of this edge is $MAST(b_1, b_2)$. Isolated vertices are not included in the graph. An edge (b_1, b_2) such that each b_i corresponds to a side child of u_i or v_i is called a *side-side* edge. If both b_i corresponds to center children, then (b_1, b_2) is a *center-center* edge. Otherwise it is respectively *side-center* or *center-side* edge. We will also use graph $B'(x_1, x_2)$ obtained from $B(x_1, x_2)$ by removing the nodes corresponding to the central children of x_1 and x_2.

Let $MM(B(x_1, x_2))$ be the maximum matching in $B(x_0, x_1)$. We have

$$MAST(x_1, x_2) = \max \begin{cases} MM(B(x_1, x_2)) \\ \max_{b_1 \in B_1} MAST(b_1, x_2) \\ \max_{b_2 \in B_2} MAST(x_1, b_2) \end{cases} \tag{4.1}$$

The above equality leads to a natural generalization of the $O(n^2)$ time dynamic programming algorithm for binary trees to an $O(n^2) + total\_MM$ algorithm for general trees where $total\_MM$ is the total cost of the matchings performed at each pair (x_1, x_2). For the trees with degree bound d there are at most n^2 matchings each of which involves a $2d$-vertex bipartite graph. Thus we have immediately an $O(n^2 d^2 \sqrt{d})$ time $MAST$ algorithm for trees with degree bound d.

To obtain more efficient algorithms the following observations are used:

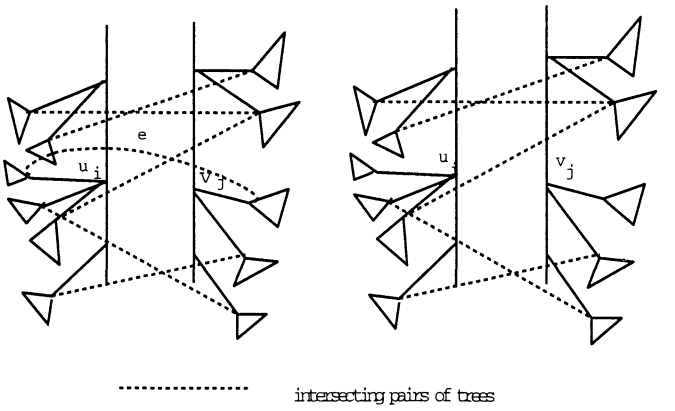


FIGURE 4. Generalization of path matching to non-binary trees

i: The bound on *total_MM* can be tighten by bounding the total number of edges in all bipartite graphs in all matchings.

ii: In each bipartite matching problem, in any set of edges (b_i, b) adjacent to the same node b and such that b_i is a vertex of degree one, we can remove all edges in this set except the heaviest one without changing the value of the maximum matching.

iii: The number of matchings can be bounded. With a natural generalization of the concept of interesting pairs, the number of matchings is bounded by the number of interesting pairs.

The first two observations and the variant of the third one are the main building blocks of the $O(n\sqrt{n}\log n)$ time algorithm of Farach and Thorup.

We define the decomposition of a tree of degree higher than two into centroid paths in the same way as in the case of binary trees: the central child is a child with largest number of descending leaves (ties are broken arbitrarily) and the remaining children are side children. The definitions of interesting pairs of paths and interesting and supporting pairs of nodes extends in the obvious way.

Using the proof technique of Lemma 2.1 we obtain:

LEMMA 4.1. *In the MAST problem for unbounded degree trees T_1, T_2 where the maximum depth of a centroid path is bounded by k, the number of all side-side edges in all bipartite graphs $B(*,*)$ is $O(kn)$.*

COROLLARY 4.2. In the *MAST* problem for unbounded degree trees T_1, T_2 the number of interesting pairs is $O(n \log n)$.

The path matching algorithm naturally generalizes to non-binary trees. Consider an interesting pair of centroid paths. Two examples of possible sets of interesting pairs are illustrated on Figure 4. The difference between the cases a) and b) is the presence or absence of the interesting pair denoted by e. The existence of e makes (u_i, v_j) interesting. In this case we will compute $MAST(u_i, v_j)$ using formula 4.1. In the second case pair (u_i, v_j) is not interesting and $MAST(u_i, v_j)$ is represented as a proper crossing.

Similarly as in the case of binary trees, we express the path matching algorithm as a matching problem on a bipartite graph $G(P_1, P_2)$. As before, there is a multiedge consisting of a white , a red, and a green edge for one pair of interesting nodes (u_i, v_i). However now the weight of a white edge equals to maximum weight matching in the graph $B'(u_i, v_j)$. We also introduce, for every interesting pair (u_i, v_i), a gray edge of weight equal to the maximum weight matching in $B(u_i, v_j)$. We may have more than one red of green edge in a multiedge (but at most one per each side subtree). As in the case of binary trees each green or red edge represents a possible agreement mapping of the corresponding side subtree to the center subtree and the weight of this edge is equal to the *MAST* value corresponding to this mapping.

Agreement Matching for non-binary trees. Let $G(P_1, P_2)$ be the bipartite graph corresponding to an interesting pair of paths (P_1, P_2). A matching in G is an agreement matching if:

1. It has zero or more white edges and at most one proper crossing or at most one gray edge.
2. No white edge crosses any other edge; further, all white edges properly dominate the edges in the proper crossing or properly dominate the gray edge if it exists.

We process all white, gray, green, and red edges in the same order as for the binary tree algorithm using similar binary search tree data structure to store values needed in the computation. The red or green edges belonging to the same multiedge are processed in an arbitrary order.

The new thing is that when processing an interesting pair (u_i, v_j) we compute two Maximum Weight Matchings: for the graphs $B(u_i, v_j)$ and $B'(u_i, v_j)$. The first value becomes the weight of the corresponding gray edge and the second value becomes the weight of the corresponding white edge. The weights of red and green edges are equal to the *MAST* values of the corresponding supporting pairs. Modulo this modification the structure of the algorithm is the same as the structure of the algorithm Agree.

We refer to this generalized algorithm as *GAgree*. Thus we have:

THEOREM 4.3. *The Algorithm GAgree can be implemented to run in* $O(n \log n) + total\_MM$ *time.*

As a corollary we obtain the result of Kao [**11**].

LEMMA 4.4. *MAST problem for degree d trees can be solved in* $O(n \log n d^2 \sqrt{d})$ *time.*

To reduce the cost further we use the idea of Farach and Thorup that allows to reduce the total number the edges in the matchings.

LEMMA 4.5. *The total number of edges in all matchings for all graphs* $B(*, *)$ *and* $B'(*, *)$ *computed by the GAgree algorithm can be reduced to* $O(n \log n)$. *Furthermore, this reduction can be carried in* $O(n \log n \log d)$ *total time.*

PROOF. Following the argument of [**5**], we estimate the number of each type of edges separately. By Lemma 4.1 there are $O(n \log n)$ side-side edges. Since there is only one center-center edge per interesting pair, the total number of center-center edges is $O(n \log n)$. The center-side edges (and the side-center edges) can be divided into two classes: The edges (c, v) such that for some side child u' the side-side edge (u', v) is in the same graph B. The number of the edges in this class is bounded

by the number of side-side edges thus it is $O(n \log n)$. All the remaining edges (c, v') except for the heaviest one can be removed without changing the cost of the maximum matching. Symmetrical argument holds for side-center edges. Thus the number of edges can be reduced to $O(n \log n)$.

To perform this reduction in $O(n \log n \log d)$ total time, we expand the data structure used in the path matching algorithm. Let $G = (L \cup R, E)$ be the bipartite graph that corresponds to a pair of central paths. For each node $v \in R$ we keep a binary tree $C(v)$ with d leaves. Each leaf corresponds to a side child of v in an arbitrary but fixed order. The tree is dynamically computed as edges adjacent to nodes corresponding to side trees of v are processed. With each leaf b_2 of this tree we associate the value $MAST(cntr(u_i), b_2)$. Each internal node has associated with it maximum of the values associated with descendant leaves. We store these values, as we process the red edges of the multigraph. A similar tree is computed for each node $u \in R$. This adds $O(n \log n \log d)$ to all instances of the Agreement Matching algorithm.

To perform the edge reduction in $B(u, v)$ we need to know the maximum over the weights of these center-side edges (c, b_i) $C(v)$ for which there does not exists a side-side edge (b_i', b_i). So far the tree C is stores the maximum over all center-side edges. We mask all b_i for which there exist such as side-side edge (b_i', b_i). To do so, we iterate over all side-side edges (b_1, b_2) in $B(u, v)$ and for each such edge we mask the value corresponding to the nodes b_2 in $C(v)$ and b_1 in $C(u)$. When we are done, the value in the root of $C(v)$ is equal to the the maximum weight over center-side edges $(cntr(v), b_2)$ such that for any $b_1 \in B_1$ (b_1, b_2) is not in B. After computing the maximum matching, we perform another iteration over all side-side edges to unmask all masked leaves. The number of masking/unmasking operations is bounded by the number of side-side edges. □

Since the total number of edges is $m = O(n \log n)$, the maximum weight is $W = n$, the number of vertices in each bipartite graph $n' = O(d)$, using the $O(m\sqrt{n'} \log W)$ algorithm of Gabow and Tarjan [**9**] we have:

COROLLARY 4.6. *MAST* problem for degree d trees can be solved in $O(n \log^2 n\sqrt{d})$ time.

For $d < \log n$ we can sharpen further this bound. Note that after applying edge reduction we have several matching problems with total number of edges $O(n \log n)$. Assume that after the edge reduction each matching problem is solved by this of algorithms of complexity $O(d^2\sqrt{d})$ and $O(m\sqrt{d} \log n)$ respectively, that gives a better complexity bound for the given number of edges. Consider all the bipartite matching problems we deal with. If any of the bipartite graphs B is not connected we count each connected component as a separate matching problem. Let m_i be the number of edges in the problem i. Then the number of vertices $n_i \leq m_i + 1$. Thus the total cost of all matchings is maximized when $m_i = d - 1$ and the number of problems is $O(n/d \log n)$. Thus we have:

COROLLARY 4.7. *MAST* problem for degree d trees can be solved in $O(n \log nd\sqrt{d})$ time.

5. An optimal algorithm for unbounded degree trees.

The first $O(\sqrt{n} \log n)$ time algorithm for unbounded degree trees was presented by Farach and Thorup. The algorithm uses the observations i-iii from the previous

section as well as the following observation. For a given constant c one can identify disjoint subproblems of size $n/2^c$, such that the solutions to these subproblems can be merged in $O(cn\sqrt{n}\log n)$ time to yield a solution to the $MAST$ problem. With properly chosen constant c this leads to a recursive algorithm of complexity $O(n\sqrt{n}\log n)$. In the description bellow, we follow the same basic approach but we replace the merging procedure with the generalized path matching algorithm. This together with some earlier lemmas, simplifies the original algorithm as well as some arguments in its analysis.

The outline of the (modified) algorithm can be described as follows:

Algorithm UAgree

Step 1: a) For every side subtree $side(x_i)$ of T_1 such that x_i belongs to a centroid path on depth c compute recursively $UAgree(side(x_i), T_2|L(side(x_i)))$.
b) For every subtree $side(x_i)$ of T_2 such that x_i belongs to a centroid path on depth c compute $BAgree(T_1|L(side(x_i)), side(x_i))$.

Step 2: Use the GAgree algorithm to compute $MAST$ value for the interesting pairs that belong to pairs of paths of depth less than c.

THEOREM 5.1. *Algorithm UAgree can be implemented to run in $O(n\sqrt{n}\log n)$ time.*

PROOF. By Lemma 4.1, there are $O(cn)$ edges in all bipartite matching problems in step 2. Thus step 2 takes $O(cn\sqrt{n}\log n)$ time. In step 1, each of the subproblems have size at most $n/2^c$. Chose $c = 4$ and let $n_1, \ldots, n_k$ and $n'_1, \ldots, n'_{k'}$ be the sizes of the subproblems in step, 1a, 1b respectively. $C(n) = O(cn\sqrt{n}\log n) + \sum_i^k C(n_i) + \sum_i^{k'} C(n'_i)$ which by a simple inductive argument is $O(n\sqrt{n}\log n)$. □

Note that a similar argument would apply if the cost of the matching was $n^{1+\epsilon}$ for some constant ϵ. Indeed, a more careful argument gives an $O(n^{1+\epsilon})$ time reduction of the $MAST$ to the Maximum Weight Bipartite Matching problem [**6**].

6. The uniform algorithm for non-binary trees.

In the previous sections we presented an $O(n\log n)$ time algorithm for binary trees and an $O(n\sqrt{n}\log n)$ algorithm for trees of unbounded degree. Bounded degree algorithms of previous sections for large d have worst complexity bound than the algorithm for unbounded degree trees. In this section, we give a new uniform $O(n\sqrt{d}\log n)$ time algorithm for computing Maximum Agreement Subtree for trees with degree bound d.

Our algorithm uses the GAgree algorithm for subproblems of small size. Using the same algorithm, it also computes $MAST(T_1|L(S), S)$ and $MAST(S, T_2|L(S))$ for some small size fragments S of T_i which formally are not rooted subtrees. The threshold value is $\log n$. For large (bigger than $\log n$) subproblems, the algorithm is a slightly modified version of GAgree.

Algorithm Agree(T_1, T_2):

1: Let P_1 be the central path of T_1 that contains the root of T_1. Using the algorithm GAgree compute $MAST(S, T_2 L(S))$ for the following subtrees S of T_2:

1a: S is a maximal subtree of size at most $\log n$ rooted at a node of the central path.

1b: S is a $\log n$ size subtree of the form $T_1(u) - cntr(u)$ where u is a node of the central path that does not belong to the subtree defined in step 1.a and $T_1(u) - cntr(u)$ the subtree rooted at it after removing the central child.

1c: S is any side tree of size at most $\log n$ that is not included in any of the subtrees above.

2: Perform step 1 with respect to the tree T_1.

3: For each side subtree Z, of T_1 of size larger than $\log n$ compute recursively AGREE$(Z, T_2|L(Z))$

4: Solve the Path Matching problem for every interesting pair of paths P_1, P_2' where P_2' is a centroid path of T_2. The order of processing these pairs agrees with the dominance partial order. Use results of steps 1 and 2 to speed up maximum matching computation (see below).

THEOREM 6.1. *The algorithm Agree can be implemented to run in* $O(n \log n \sqrt{d})$ *time for* $d \geq \log n$ *and in* $O(n \log n \log d + nd\sqrt{d})$ *time for* $d < \log n$.

PROOF. By Lemma 4.7, the complexity of the first two steps is $O(n(\log\log n)\sqrt{\log n}) = O(n \log n)$ time. The time cost of the remaining steps of the algorithm is dominated by the maximum of $O(n \log n)$ and the cost of computing all remaining matchings.

A node x_i such that $size(T_i(x_i) - cntr(x_i)) > \log n$ is called a *branching node.* A side node (a side subtree) such that the side subtree rooted at it has size at most $\log n$ is called *smallside* node (subtree).

First, we identify matchings that can be computed in constant time. If (x_1, x_2) is an interesting pair such that none of x_i is a branching node, then pair (x_1, x_2) has been processed in steps 1 and 2. The values $MM(B(x_1, x_2))$ and $MM(B'(x_1, x_2))$ can be computed in constant time from the results of these steps. Namely, $MM(B'(x_1, x_2))$ is equal to the $MM(B'(x_1, x_2))$ computed for the pair (x_1, x_2) in steps 1 and 2. The maximum matching for the graph $B(x_1, x_2)$ is the heavier of the following matchings: $MM(B'(x_1, x_2)) \cup (cntr(x_1), cntr(x_2))$ and the matching for the graph B computed for the pair (x_1, x_2) in steps 1 and 2.

Thus, we can restrict our attention to these matching problems $MM(x_1, x_2)$ where at least one of x_i is a branching node.

By Lemma 4.5, it suffices to count the number of side-side edges in all these matchings. We divide these edges into classes: class *smallside-smallside* contains edges whose both endpoints correspond to smallside subtrees *branching-smallside*, *smallside-branching*, and *branching-branching.* In each case term "branching" stands for an endpoint associated with a node that contains a branching node in its side subtree.

Since outside of steps 1 and 2, the algorithm does not recurse on smallside subtrees and all smallside subtrees are disjoint thus the number of smallside-smallside edges is bounded by n.

To bound the number of branching-branching edges, we use an argument similar to the argument used in the proof of the Lemma 2.1. A *singleton* edge of a given type is an edge such that none of its endpoints is incident to another edge of the same type. First note that there are $O(n)$ singleton branching-branching edges. This follows from the observation that the number of singleton branching-branching

edges in a given recursive step is bounded by $n/\log n$. (It is bounded by the number of leaves of the tree obtained form T_2 after removing all maximal subtrees of size less than $\log n$).

To bound non-singleton edges, first identify among these edges *fat branching-branching* edges defined to be branching-branching edges defined by a pair of side trees whose intersection contains at least $\log n$ nodes.

Subsequently define a *left-nonsingleton fat branching-branching* edge as a fat branching-branching edge (x_1, x_2) for which there exists another fat branching-branching edge (x_1, x_2') such that x_2' belongs to the same central path as x_2. Let u_i be a branching node and m_i be the size of the side tree rooted at this node. Then there is $O(m_i/\log n)$ fat left non-singleton branching-branching edges, where the left endpoint is u_i. Thus over all branching nodes in P_i there is $O(n/\log n)$ such edges and summing over all recursive steps gives $O(n)$ left-nonsingleton fat branching-branching edges. The argument for right non-singleton fat branching-branching edges is similar.

Now we move to non-fat left non-singleton branching-branching edges. There are at most m_i non-fat left nonsingleton branching-branching edges (u_i, v) whose left endpoint is u_i. Since in the next recursive step a successor of u_i and a successor of v cannot be endpoints of a branching-branching edge the total number of left non-singleton non-fat branching branching edges is also $O(n)$.

It remains to count smallside-branching edges. (The argument for branching-smallside edges is symmetric). With respect to these edges we define the left non-singleton smallside-branching edges in the standard way. Since, in step 3, the algorithm does not recurse on smallside trees, there is $O(n)$ such left non-singleton smallside-branching edges.

To bound the number of left-singleton edges, note that the number of left-singleton smallside-branching edges whose left endpoint is incident to an edge of another type is bounded by the number of edges of other types and thus is $O(n)$. Thus we can concentrate on the rest of the edges. Note, that all of this remaining edges that belong to the same merging problem for a graph $B(x_0, x_1) = (B_1 \cup B_2, E)$ and are incident the same node in B_2, can be replaced by that one of them that has biggest weight (observation ii in Section 4). Thus, we can assume that each branching vertex in B_2 has at most one such remaining left singleton edge incident to it. Further, since we need to count only the number of edges in the matching problems that cannot be solved in constant time, we can restrict our attention to these matching problems that have at least two smallside-branching or branching-smallside edges. Assume that there are two or more left singleton smallside-branching edges in $MM(x_1, x_2)$. Using the same counting argument as in the case of left-nonsingleton edges we obtain that the number of such pairs is $O(n)$. Similar argument applies when we have more than one branching-smallside edge.

We conclude that the number of edges in non-constant time matching problems can be reduced to linear. The cost of the reduction adds $O(\log d)$ per each processed edge. Thus the total cost of the algorithm is $O(n \log n \log d + n \log n\sqrt{d}) = O(n \log n\sqrt{d})$, or using argument of Lemma 4.7 we have alternative bound of $O(n \log n \log d + nd\sqrt{d})$. □

7. Conclusions

In this paper we discussed subquadratic algorithms for the $MAST$ problem. We presented these results using a uniform framework that combines the techniques of several papers [**1, 3, 2, 5, 6**].

We presented an $O(n \log n\sqrt{d})$ algorithm for computing $MAST$ problem for trees of degree bounded by d. In the case when $d < \log n$ the algorithm takes $O(n \log n \log d + nd\sqrt{d})$ time. Since for binary trees, the $MAST$ problem is at least as hard as the longest common subsequence problem and for trees with degree bound d it is at least as hard as the maximum matching in a bipartite graph with $O(d)$ vertices, the upper bound for the $MAST$ problem corresponds to the upper bound for the respective problem.

The same sparsification techniques can be used for the $MAST$ problem for unrooted trees of bounded degree, leading to an $O(n \log n)$ time algorithm [**16**].

References

[1] R. Cole, M. Farach, R. Hairhanan, T.M.Przytycka, and M. Thorup. An $O(n \log n)$ time algorithm for the Maximum Agreement Subtree Problem for Binary trees. Journal version of [2] and [4], in preparation.

[2] R. Cole and R. Hairhanan An $O(n \log n)$ time algorithm for the Maximum Agreement Subtree Problem for Binary trees. *Proc. of the 7th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 323–332, 1996. *SODA 96* pages 323–332, 1996.

[3] M. Farach, T. M. Przytycka, and M. Thorup. Agreement of bounded degree trees. *ESA95* pp. 381-394.

[4] M. Farach, T. M. Przytycka, and M. Thorup. On the Agreement of Many Trees. *IPL* 55, pp 297-301 (1995).

[5] M. Farach and M. Thorup. Fast comparison of evolutionary trees (extended abstract). *Proc. of the 5th Annual ACM-SIAM Symposium on Discrete Algorithms*, *SODA94* pages 481–488, 1994.

[6] M. Farach and M. Thorup. Sparse dynamic programming for evolutionary tree comparison. *Proc. of the 35th IEEE Annual Symp. on Foundation of Computer Science*, *FOCS94* pages 770–779, 1994.

[7] J. Felsenstein. Numerical methods for inferring evolutionary trees. *The Quarterly Review of Biology*, 57(4), 1982.

[8] C. R. Finden and A. D. Gordon. Obtaining common pruned trees. *Journal of Classification*, 2: 255–276, 1985.

[9] H. Gabow and R. Tarjan. Faster scaling algorithms for network problems. *SIAM Journal on Computing*, 18(5), 1013–1036, 1989.

[10] G. Jacobson and K-P. Vo. Heaviest increasing/common subsequence problems. In *3rd Symposium on Combinatorial Pattern Matching*, 1992.

[11] M-Y. Kao. Tree contractions and evolutionary trees. To appear in SIAM Journal of Computing, 1995.

[12] E. Kubicka, G. Kubicki, and F. R. McMorris. An algorithm to find agreement subtrees. *Journal of Classification*,91–99, 1995.

[13] W. Goddard, E. Kubicka, and F. R. McMorris. The Agreement Metric for Labeled Binary Trees. *Mathematical Biosciences* 123, 225–336, 1994.

[14] T. W. Lam, W. K. Sung, H. F. Ting. Computing the Unrooted Maximum Agreement Subtree in Sub-quadratic time. *Proc. of SWAT'96* LNCS 1097, 124–135, 1996.

[15] M. Steel and T. Warnow. Kaikoura tree theorems: Computing the maximum agreement subtree. *Information Processing Letters*, 48, 77–82, 1993.

[16] T. M. Przytycka. Reducing rooted agreement to unrooted agreement. Submitted.

JOHNS HOPKINS UNIVERSITY, SCHOOL OF MEDICINE, 725 NORTH WOLF STREET, BALTIMORE, MD21205

E-mail address: przytyck@grserv.med.jhmi.edu

https://doi.org/10.1090/dimacs/037/18

DIMACS Series in Discrete Mathematics
and Theoretical Computer Science
Volume **37**, 1997

The Median Function on Weak Hierarchies

F. R. McMorris and R. C. Powers

ABSTRACT. Extending previous work on hierarchies, a characterization of the median function on the set of weak hierarchies is given.

1. Introduction

There are many ways to construct a classification scheme for a finite set of entities S, and there are several forms that such a classification may take. However, most classification schemes will have one aspect in common: each consists of a set of non-empty subsets of S called *clusters*. Thus, in mathematical terminology, the central part of any classification is a simple *hypergraph* on S (a set of non-empty subsets of S) with the clusters being the edges and S the set of vertices. Of course classification hypergraphs have additional restrictions. One of the most popular is that of being a hierarchy. A hypergraph H on S is a *hierarchy* if and only if $S \in H$, $\{x\} \in H$ for all $x \in S$, and $A \cap B \in \{A, B, \emptyset\}$ for all $A, B \in H$.

Generalizing hierarchies to allow for partial overlap of clusters, Bandelt and Dress [**1**] introduced weak hierarchies. A hypergraph H is a *weak hierarchy* if and only if $S \in H$, $\{x\} \in H$ for all $x \in S$, and $A \cap B \cap C \in \{A \cap B, A \cap C, B \cap C\}$ for all $A, B, C \in H$. Note that if $A \cap B$, $A \cap C$, $B \cap C$ are nonempty sets, then $A \cap B \cap C$ is a nonempty set. Clearly every hierarchy is a weak hierarchy but not conversely.

We are concerned with the problem of forming the consensus of several hierarchies or weak hierarchies that may, for example, have been constructed by applying different algorithms to the same data set. One method of consensus, the median function, returns those weak hierarchies that are "close" to the input hierarchies. Before recalling the definition of this method, we must first also recall some other basic terminology and definitions.

For any set X, let $P(X)$ be the set of all subsets of X. Let $\mathcal{W}$ be the set of all weak hierarchies on S and d the symmetric difference metric on $P(P(S))$. That is, for $H_1, H_2 \in P(P(S))$, $d(H_1, H_2) = |H_1| + |H_2| - 2|H_1 \cap H_2|$. Set $\mathcal{W}^* = \cup_{k>0}\mathcal{W}^k$ where $\mathcal{W}^k$ is the k-fold cartesian product of $\mathcal{W}$, and call elements of $\mathcal{W}^*$ *profiles*. For a profile $P = (H_1, ..., H_k)$ and $H \in P(P(S))$ we set $D(H, P) = \sum_{i=1}^{k} d(H, H_i)$. The *median function* on $\mathcal{W}$ is the function $m : \mathcal{W}^* \to P(\mathcal{W}) \setminus \emptyset$ defined by $m(P) = \{H \in \mathcal{W}\colon D(H, P) \text{ is minimum}\}$. In this paper we characterize m by building on previous work characterizing the median function on hierarchies [**3**]. We note that the material reported here can be generalized to order theoretic results along the

1991 *Mathematics Subject Classification.* Primary 92B10; Secondary 05C65.
This research was supported in part by ONR grant N00014-95-1-0109 to F. R. McMorris.

lines of [2], [5] and [6], and this will be reported elsewhere. In particular we can show that Theorem 10 extends to an arbitrary finite distributive semilattice.

2. The Result

We will characterize m among the *consensus functions* on $\mathcal{W}$, which are simply mappings $c : \mathcal{W}^* \to P(\mathcal{W}) \setminus \emptyset$. We start by stating some standard properties for a consensus function c. If $c((H)) = \{H\}$ for all $H \in \mathcal{W}$, then c is *faithful.* Next, c is *consistent* if for all profiles P and P', the condition $c(P) \cap c(P') \neq \emptyset$ implies that $c(PP') = c(P) \cap c(P')$, where PP' denotes the concatenation of the two profiles P and P'. Letting $(H)^k \in \mathcal{W}^k$ be the "constant" profile $(H)^k = (H, ..., H)$, c satisfies *unanimity* if $c((H)^k) = \{H\}$ for all $H \in \mathcal{W}$ and $k > 0$. It is easy to see that if c is faithful and consistent, then c satisfies unanimity.

It is routine to show the following ([2], [3]).

LEMMA 1. *The median function m is faithful and consistent.*

Let $A \subseteq S$ and $P = (H_1, ..., H_k) \in \mathcal{W}^*$. The *index* of A in P is

$$\gamma(A, P) = \frac{|\{i : A \in H_i\}|}{k}.$$

The following formula will be useful in the sequel.

LEMMA 2. *Let $A \subseteq S$, $P = (H_1, ..., H_k) \in \mathcal{W}^*$, and $H \subseteq P(S)$. If $A \notin H$, then*

$$D(H \cup \{A\}, P) = D(H, P) + k(1 - 2\gamma(A, P)).$$

PROOF. If $A \in H_i$, then $d(H \cup \{A\}, H_i) = d(H, H_i) - 1$. If $A \notin H_i$, then $d(H \cup \{A\}, H_i) = d(H, H_i) + 1$. So $D(H \cup \{A\}, P) = \sum_{i=1}^{k} d(H \cup \{A\}, H_i) = \sum_{\{i:A \in H_i\}} (d(H, H_i) - 1) + \sum_{\{i:A \notin H_i\}} (d(H, H_i) + 1) = \sum_{\{i:A \in H_i\}} d(H, H_i) - k\gamma(A, P) + \sum_{\{i:A \notin H_i\}} d(H, H_i) + (k - k\gamma(A, P)) = \sum_{i=1}^{k} d(H, H_i) + k(1 - 2\gamma(A, P)) = D(H, P) + k(1 - 2\gamma(A, P))$. □

A consensus function c is $\frac{1}{2}$-*condorcet* if, for any $A \subseteq S$ and $P = (H_1, ..., H_k) \in \mathcal{W}^*$ such that $\gamma(A, P) = \frac{1}{2}$, $H \in c(P)$ if and only if $H \cup \{A\} \in c(P)$ provided $H \cup \{A\} \in \mathcal{W}$. The next result is an immediate consequence of Lemma 2.

LEMMA 3. *The median function m is $\frac{1}{2}$-condorcet.*

In [6], some results of [2] and [3] were improved to show that a consensus function defined for hierarchies is the median function on hierarchies if and only if it is faithful, consistent and $\frac{1}{2}$-condorcet. For weak hierarchies, these three properties are not sufficient to characterize m as the next example illustrates.

EXAMPLE 4. *Define a consensus function c on $\mathcal{W}$ as follows:*

$$c(P) = \{H : D'(H, P) \text{ is maximum}\}$$

where $D' : \mathcal{W} \times \mathcal{W}^* \to \Re$ is given by

$$D'(H, P) = \sum_{\{A \in H : 1 < |A| < n\}} |A| k(2\gamma(A, P) - 1).$$

Here k is the length of the profile P.

It is shown in [6] that c is faithful, consistent, and $\frac{1}{2}$-condorcet but that $c \neq m$, when $n = 8$ for example.

For each profile $P = (H_1, ..., H_k) \in \mathcal{W}^*$, define a weight function $w(P) = w : P(S) \to \Re$ by

$$w(A) = k(2\gamma(A, P) - 1)$$

and let

$$J_{\frac{1}{2}}(P) = \{A \in P(S) : \gamma(A, P) \geq \frac{1}{2}\}.$$

Note that $w(A) \geq 0$ if and only if $A \in J_{\frac{1}{2}}(P)$. Similar weighting functions were defined in a slightly different context in [4].

LEMMA 5. *Let $P = (H_1, ..., H_k) \in \mathcal{W}^*$. If $H \subseteq J_{\frac{1}{2}}(P)$, then*

$$D(H, P) = D(J_{\frac{1}{2}}(P), P) + \sum_{\{A \in J_{\frac{1}{2}}(P) : A \notin H\}} w(A).$$

PROOF. If we use the function w, then the formula given in the statement of Lemma 2 becomes: $D(H \cup \{A\}, P) = D(H, P) - w(A)$. By repeated application of Lemma 2 we get $D(H \cup \{A_1, ..., A_l\}, P) = D(H, P) - \sum_{i=1}^{l} w(A_i)$ where $H \cup \{A_1, ..., A_i\}$ is a proper subset of $H \cup \{A_1, ..., A_{i+1}\}$ for $i = 1, .., l-1$. In particular, since $H \subseteq J_{\frac{1}{2}}(P)$, we get $D(J_{\frac{1}{2}}(P), P) = D(H, P) - \sum_{\{A \in J_{\frac{1}{2}}(P) : A \notin H\}} w(A)$. □

For each $H \in \mathcal{W}$ let

$$\overline{w}_p(H) = \sum_{\{A \in J_{\frac{1}{2}}(P) : A \notin H\}} w(A)$$

and when $H \supseteq J_{\frac{1}{2}}(P)$ we set $\overline{w}_p(H) = 0$. A consensus function c is $\frac{1}{2}$-*weighted* if, for any profile $P \in \mathcal{W}^*$, $c(P) \subseteq \overline{w}(P) = \{H \in \mathcal{W}: \overline{w}_p(H)$ is minimized $\}$.

LEMMA 6. *For any $H \in \mathcal{W}$, $\overline{w}_p(H) = \overline{w}_p(H \cap J_{\frac{1}{2}}(P))$.*

PROOF. Let $H \in \mathcal{W}$. Then $\{A \in J_{\frac{1}{2}}(P) : A \in H\} = \{A \in J_{\frac{1}{2}}(P) : A \in H \cap J_{\frac{1}{2}}(P)\}$. So $\{A \in J_{\frac{1}{2}}(P) : A \notin H\} = \{A \in J_{\frac{1}{2}}(P) : A \notin H \cap J_{\frac{1}{2}}(P)\}$. Thus $\overline{w}_p(H) = \overline{w}_p(H \cap J_{\frac{1}{2}}(P))$. □

LEMMA 7. *The median function m is $\frac{1}{2}$-weighted.*

PROOF. Let $P = (H_1, ..., H_k) \in \mathcal{W}^*$ and $H \in m(P)$. Let $A \in H$. If $H' = H - \{A\}$, then $H = H' \cup \{A\}$ and, by Lemma 2, $D(H, P) = D(H', P) - w(A)$. Since $H \in m(P)$ it follows that $w(A) \geq 0$. So $\gamma(A, P) \geq 0$. Thus $H \subseteq J_{\frac{1}{2}}(P)$. It follows from Lemma 5 that $D(H, P) = D(J_{\frac{1}{2}}(P), P) + \overline{w}_p(H)$. So $\overline{w}_p(H) = D(H, P) - D(J_{\frac{1}{2}}(P), P)$. Let $H'' \in \mathcal{W}$. Then, since $H \in m(P)$, $D(H'' \cap J_{\frac{1}{2}}(P), P) \geq D(H, P)$. By Lemmas 5 and 6, $\overline{w}_p(H) = D(H, P) - D(J_{\frac{1}{2}}(P), P) \leq D(H'' \cap J_{\frac{1}{2}}(P), P) - D(J_{\frac{1}{2}}(P), P) = \overline{w}_p(H'' \cap J_{\frac{1}{2}}(P)) = \overline{w}_p(H'')$. So $\overline{w}_p(H) \leq \overline{w}_p(H'')$ for any $H'' \in \mathcal{W}$. Thus $H \in \overline{w}(P)$. It follows that $m(P) \subseteq \overline{w}(P)$. Hence m is $\frac{1}{2}$-weighted. □

LEMMA 8. *Suppose c is a consensus function on $\mathcal{W}$ that satisfies faithfulness, cosistency and $\frac{1}{2}$-condorcet. For any profile $P = (H_1, ..., H_k) \in \mathcal{W}^*$ and $H \in c(P)$, $H \subseteq J_{\frac{1}{2}}(P)$.*

PROOF. If $k = 1$, then the result follows from faithfulness. So we can assume that $k \geq 2$. Suppose that there exist $H \in c(P)$ and $A \subseteq S$ such that $H = H' \cup \{A\}$, with $H' \neq H$ and $\gamma(A, P) < \frac{1}{2}$.

Now suppose that $\gamma(A, P) = \frac{u}{k} < \frac{1}{2}$. Let $P^* = P(H)^{k-2u}$. It follows from faithfulness and consistency that $c(P^*) = \{H\}$. Note that $\gamma(A, P^*) = \frac{u+(k-2u)}{2k-2u} = \frac{1}{2}$. It follows from $\frac{1}{2}$-condorcet that $H' \in c(P^*)$, which contradicts $c(P^*) = \{H\}$. □

Our next aim is to extend Lemma 5. Towards this end, for each $H \in \mathcal{W}$ and $P \in \mathcal{W}^*$ let

$$\underline{w}_p(H) = \sum_{\{A \in H : A \notin J_{\frac{1}{2}}(P)\}} w(A)$$

and when $H \subseteq J_{\frac{1}{2}}(P)$ we set $\underline{w}_p(H) = 0$. Since $w(A) < 0$ whenever $A \notin J_{\frac{1}{2}}(P)$ it follows that $\underline{w}_p(H) \leq 0$ for all $H \in \mathcal{W}$.

LEMMA 9. *Let $P = (H_1, ..., H_k) \in \mathcal{W}^*$ and $H \in \mathcal{W}$. Then*

$$D(H, P) = D(J_{\frac{1}{2}}(P), P) + \overline{w}_p(H) - \underline{w}_p(H).$$

PROOF. Let $H \in \mathcal{W}$ and set $H' = H \cap J_{\frac{1}{2}}(P)$. By Lemma 5, $D(H', P) = D(J_{\frac{1}{2}}(P), P) + \overline{w}_p(H')$. Now $H = H' \cup \{A \in H : A \notin J_{\frac{1}{2}}(P)\}$. By repeated application of Lemma 2 we get $D(H, P) = D(H', P) - \underline{w}_p(H)$. Hence $D(H, P) = D(J_{\frac{1}{2}}(P), P) + \overline{w}_p(H) - \underline{w}_p(H)$. □

For the next result note that any set of consensus functions on $\mathcal{W}$ can be ordered pointwise, i.e., $c_1 \leq c_2$ if and only if $c_1(P) \subseteq c_2(P)$ for all profiles P.

THEOREM 10. *The median function m on $\mathcal{W}$ is the maximum element of the set of all consensus functions on $\mathcal{W}$ which are faithful, consistent, $\frac{1}{2}$-condorcet, and $\frac{1}{2}$-weighted.*

PROOF. It follows from Lemmas 1,3, and 7 that m is faithful, consistent, $\frac{1}{2}$-condorcet, and $\frac{1}{2}$-weighted.

Suppose that the consensus function c on $\mathcal{W}$ is faithful, consistent, $\frac{1}{2}$-condorcet, and $\frac{1}{2}$-weighted. It follows from Lemma 8 that $H \subseteq J_{\frac{1}{2}}(P)$ for all $P \in \mathcal{W}^*$ and $H \in c(P)$. Since c is $\frac{1}{2}$-weighted, $\overline{w}_p(H) \leq \overline{w}_p(H')$ for any $H' \in \mathcal{W}$. Now, by Lemmas 5 and 9, $D(H, P) = D(J_{\frac{1}{2}}(P), P) + \overline{w}_p(H) \leq D(J_{\frac{1}{2}}(P), P) + \overline{w}_p(H') \leq D(J_{\frac{1}{2}}(P), P) + \overline{w}_p(H') - \underline{w}_p(H') = D(H', P)$ for all $H' \in \mathcal{W}$ and $P \in \mathcal{W}^*$. So $c(P) \subseteq m(P)$. Since P was arbitrary it follows that $c \leq m$. Hence m is the maximum element of the set of all consensus functions on $\mathcal{W}$ which are faithful, consistent, $\frac{1}{2}$-condorcet, and $\frac{1}{2}$-weighted. □

We conclude this paper with the following open problem. Characterize the class of consensus functions on $\mathcal{W}$ that satisfy the axioms of faithfulness, consistency and $\frac{1}{2}$-condorcet.

References

[1] H.-J. Bandelt and A. W. M. Dress, *Weak hierarchies associated with similarity measures-an additive clustering technique*, Bull. Math. Biol. **43** (1989), 133-166.

[2] J. P. Barthélemy and M. F. Janowitz, *A formal theory of consensus*, SIAM J. Discrete Math. **4** (1991), 305-322.

[3] J. P. Barthélemy and F. R. McMorris, *The median procedure for n-trees*, Journal of Classification **3** (1986), 329-334.

[4] J. P. Barthélemy and B. Monjardet, *The median procedure in cluster analysis and social choice theory*, Math. Soc. Sci. **1** (1981), 235-267.

[5] F. R. McMorris, H. M. Mulder and R. C. Powers, *The median function on median graphs and median semilattices*, submitted (1996).

[6] F. R. McMorris and R. C. Powers, *The median procedure in a formal theory of consensus*, SIAM J. Discrete Math. **14** (1995), 507-516.

Department of Mathematics, University of Louisville, Louisville, KY 40292, USA
E-mail address: frmcmo01@homer.louisville.edu

Department of Mathematics, University of Louisville, Louisville, KY 40292, USA
E-mail address: rcpowe01@ulkyvx.louisville.edu

This page intentionally left blank

https://doi.org/10.1090/dimacs/037/19

DIMACS Series in Discrete Mathematics
and Theoretical Computer Science
Volume **37**, 1997

Towards a Theory of Holistic Clustering

Andreas W.M. Dress

ABSTRACT. In this note, cluster theory is presented from a rather abstract point of view, basic known results are reviewed from this view point, and some new results which motivated the proposed approach, as well as some new problems which naturally arise in this context, are presented.

1. A Local-Global Approach to Clustering

Clustering procedures (as well as many other mathematical techniques) can be viewed as methods for extracting globally relevant features from locally distributed information. A rather natural, simple and sufficiently general conceptual framework for describing clustering procedures is, therefore, the following one: We assume that, for any given (generally finite) set X, we can form the set $Inf\,(X)$ comprising **all** possible structures defined on X which encode the information regarding X we are seeking: this could be the set of all tree structures or the set of all ultrametrics definable on X, or just the set $\mathcal{P}(\mathcal{P}(X))$ of all subsets of the power set $\mathcal{P}(X)$ of X. In addition, we assume that, for any pair (X,Y) consisting of a set X and a subset Y of X, information regarding X implies information regarding Y which is expressed in form of a map

$$res \;=\; res_{X\to Y} : Inf\,(X) \to Inf(Y) : i \mapsto i|_Y$$

called the *restriction map* (relative to X and Y), and we assume consistency of restriction by requiring that, for all $Z \subseteq Y \subseteq X$ and $i \in \mathrm{Inf}(X)$, we have

$$i|_Z = (i|_Y)|_Z \ .$$

Finally, we assume that there is a concept of *compatibility* of information, expressed for each set X as above in terms of a - not necessarily symmetric - binary relation

$$Cpb_X \subseteq\; Inf_X \times\; Inf_X$$

- with $(i,j) \in\; Cpb_X$ implying that given information "i", the information "j" is compatible with (or even implied by) "i". A natural requirement for these compatibility relations is, of course, that all these relations are *reflexive* and *hereditary*

1991 *Mathematics Subject Classification.* 06A07, 92G30, 92B10.

with respect to restriction, that is, one has $(i,i) \in Cpb_X$ for all X and $i \in Inf(X)$ as well as

$$(i|_Y, j|_Y) \in Cpb_Y$$

for all $Y \subseteq X$ and $(i,j) \in Cpb_X$. Another property one may or may not want to require is transitivity, that is that $(i,j),(j,k) \in Cpb_X$ for some X always implies $(i,k) \in Cpb_X$.

Now, locally distributed information can be expressed in terms of elements $i_Y \in Inf(Y)$ - with Y running through the set $\mathcal{Y} \subseteq \mathcal{P}(X)$ of all those - generally small - subsets Y of X for which certified information $i_Y \in Inf(Y)$ is available and, given such a family $(i_Y)_{Y \in \mathcal{Y}}$ of certified local information, we may ask for all elements $i_X \in Inf(X)$ with $(i_Y, i_X|_Y) \in Cpb(Y)$ for all $Y \in \mathcal{Y}$.

This scheme of formalizing clustering procedures has the advantage that it allows for the separation of various stages of clustering, in particular it allows one to clearly separate the task of setting up an appropriate formal model for the kind of information we are seeking from (a) analysing the model from a mathematical point of view and (b) designing and analysing algorithms for actually computing globally relevant information from local data.

In this note, we'll study in particular the following two basic clustering models which we'll dub the *affine* and the *projective* clustering model: In the affine clustering model, the information we seek is gathered in terms of a collection $\mathcal{C}$ of *clusters*, that is, of subsets C of the set X of objects in question; so we have

$$Inf(X) = Cl^{aff}(X) := \mathcal{P}(\mathcal{P}(X)).$$

There are several ways to define, for any pair (X,Y) with $Y \subseteq X$, the associated restriction map. The most natural one probably is using intersection with Y, that is putting

$$res_{X \to Y}(\mathcal{C}) = del_{X \to Y} = del^{aff}_{X \to Y} := \{Y \cap C \mid C \in \mathcal{C}\}$$

for any $\mathcal{C} \subseteq \mathcal{P}(X)$, also sometimes called the *deletion operator* because elements outside Y are just neglected and, henceforth, deleted. Another, sometimes interesting and useful choice is using the *contraction operator*, defined by

$$contract_{X \to Y}(\mathcal{C}) = contract^{aff}_{X \to Y}(\mathcal{C}) := \{C \subseteq Y \mid C \cup (X - Y) \in \mathcal{C}\},$$

or its *dual*, the *focus operator*, defined by

$$focus_{X \to Y}(\mathcal{C}) = focus^{aff}_{X \to Y}(\mathcal{C}) := \{C \subseteq Y \mid C \in \mathcal{C}\}$$

which focuses attention to only those clusters $C \in \mathcal{C}$ which are already contained in Y. It is clear that these three operators satisfy the consistency condition described above. In this note, we'll only consider the deletion operator when dealing with Cl^{aff}.

Finally, at least when dealing with Cl^{aff}, we define the binary relation

$$Cpb_X \subseteq Inf(X) \times Inf(X)$$

by

$$Cpb_X = Cpb^{aff}_X := \{(\mathcal{C}, \mathcal{C}') \in Inf(X)^2 \mid \mathcal{C} \supseteq \mathcal{C}'\}.$$

The projective clustering model is defined in a rather similar way except that we replace the subsets C of X by the splits S of X, that is unordered pairs $S = \{A, B\}$ of subsets of X with $A \cup B = X$ and $A \cap B = \emptyset$ or - equivalently - equivalence relations $\overset{S}{\sim}$ defined on X with, at most, two distinct equivalence classes. So, with $\mathcal{S}p(X)$ denoting the set of all splits of X, we put

$$Cl^{proj}(X) := \mathcal{P}(\mathcal{S}p(X)),$$

we define

$$res_{X\to Y}(\mathcal{S}) = del_{X\to Y}(\mathcal{S}) = del^{proj}_{X\to Y}(\mathcal{S}) := \{\{A \cap Y, B \cap Y\} \mid \{A, B\} \in \mathcal{S}\}$$

for all $Y \subseteq X$ and $\mathcal{S} \subseteq \mathcal{S}p(X)$, and we define

$$Cpb^{proj}_X := \{(\mathcal{S}, \mathcal{S}') \in \mathcal{S}p(X)^2 \mid \mathcal{S} \supseteq \mathcal{S}'\}.$$

Of course, one could also define a restriction operator which is analogous simultaneously to the contraction and the focus operator by putting

$$contract_{X\to Y}(\mathcal{S}) = contract^{proj}_{X\to Y}(\mathcal{S}) := \{\{A, B\} \in \mathcal{S}p(Y) \mid \{A, B \cup (X - Y)\} \in \mathcal{S}\};$$

yet, we will not study this restriction operator in the present note and just mention in passing that in the special case $\#(X - Y) = 1$, both restriction operators, projective deletion and projective contraction, coincide.

Note also that in both set-ups the non-uniqueness of the solution i_X we are seeking for any given some family $(i_Y)_{Y\in\mathcal{Y}}(\mathcal{Y} \subseteq \mathcal{P}(X))$ is not a problem as - almost by definition - there exists always a unique largest element $\bar{i}_X \in Cl^{aff/proj}(X)$ such that i_X is a solution if and only if $(\bar{i}_X, i_X) \in Cpb^{aff/proj}(X)$.

These models formalize the following idea: Given a collection X of objects under consideration, we seek to specify "relevant" subsets C (or splits $S = \{A, B\}$) of X which group together elements of X which exhibit a certain degree of similarity relative to each other (or share enough common features) so as to distinguish them clearly, as a class C (or A, or B), from all elements outside this class. We assume further that for (at least some) small subsets Y of X, the data we can start with allow us to specify easily and directly those subsets (or splits) of Y which are considered to be relevant as far as only objects from Y are concerned, and we then ask for those subsets C (or splits S) of X which, when restricted to any such $Y \subseteq X$, produce only such subsets (or splits) of Y which have been specified before as being of some relevance relative to Y.

Of course, there are simple relations between affine and projective clustering: Given a system $\mathcal{C} \subseteq \mathcal{P}(X)$ of clusters, we can associate to it a system $\mathcal{S}$ of splits of X by putting

$$\mathcal{S} = \mathcal{S}(\mathcal{C}) := \{\{C, X - C\} \mid C \in \mathcal{C}\},$$

and given a system $\mathcal{S} \subseteq \mathcal{S}p(X)$ of splits of X, we can associate to it the system

$$\mathcal{C} = \mathcal{C}(\mathcal{S}) := \{C \subseteq X \mid \{C, X - C\} \in \mathcal{S}\}.$$

Clearly, we have

$$\mathcal{S}(\mathcal{C}(\mathcal{S})) = \mathcal{S}$$

and

$$\mathcal{C}(\mathcal{S}(\mathcal{C})) = \{C \subseteq X \mid C \in \mathcal{C} \text{ or } X - C \in \mathcal{S}\}.$$

Yet, there is a better way to relate affine and projective clustering which also explains why the terms *affine* and *projective* were suggested in this context: Given a set X as above and a system $\mathcal{C}$ of clusters $C \subseteq X$, we may add to X another *ideal* element *at infinity* denoted by $*$, and then form the split system $\mathcal{S}^*(\mathcal{C})$ of $X^* := X \cup \{*\}$, defined by

$$\mathcal{S}^*(\mathcal{C}) := \{\{C, X^* - C\} \mid C \in \mathcal{C}\},$$

while given a system $\mathcal{S}$ of splits of X^*, we may form the system $\mathcal{C}_*(\mathcal{S})$ of clusters in X defined by

$$\mathcal{C}_*(\mathcal{S}) := \{C \subseteq X \mid \{C, X^* - C\} \in \mathcal{S}\}.$$

Clearly, we have $\mathcal{S}^*(\mathcal{C}_*(\mathcal{S})) = \mathcal{S}$ for every $\mathcal{S} \subseteq \mathcal{S}p(X^*)$ as well as $\mathcal{C}_*(\mathcal{S}^*(\mathcal{C})) = \mathcal{C}$ for every $\mathcal{C} \subseteq \mathcal{P}(X)$. We'll also see later on that relevant properties of cluster systems $\mathcal{C} \subseteq \mathcal{P}(X)$ easily translate into corresponding properties of split systems $\mathcal{S} \subseteq \mathcal{S}p(X^*)$, and that similar results then hold in both situations. Moreover, as in geometry, it will turn out that while the affine version is more easily grasped and reflects the naive intuitive understanding of clustering, the projective version (which, from the affine point of view, consists in forgetting the special role of the point at infinity used in forming splits from clusters) allows one often more elegant formulations of theorems and proofs.

Regarding the above set up, the following problems arise:

(1) Given some data regarding the elements of X, e.g. a - perhaps only partially known - (dis)similarity matrix, and some type of problem, e.g. the problem of (re)constructing from these data the topology of a (phylogenetic) tree, what is the appropriate choice for the set $\mathcal{Y}$ of "small" subsets and which definition do we use to express the local information we have in terms of an element $i_Y \in Inf(Y) = Cl^{aff/proj}(Y)$?

(2) Given a family $(i_Y)_{Y \in \mathcal{Y}}$ of local data, what is the optimal algorithm for computing the associated unique largest solution

$$\bar{i}_X \in Inf(X) = Cl^{aff/proj}(X)?$$

(3) And, closely related to that question, are there *a priori* upper bounds regarding the number of clusters or splits in $\bar{i}_X$?

(4) In addition, the following questions are of interest also in the general context outlined above:

- given $\mathcal{Y} \subseteq \mathcal{P}(X)$ and $i \in Inf(X)$, what can be said about all $i_X \in Inf(X)$ with $(i|_Y, i_X|_Y) \in Cpb(Y)$ for all $Y \in \mathcal{Y}$? Obviously, this is the case for all $i_X \in Inf(X)$ with $(i, i_X) \in Cpb(X)$ whenever compatibility is hereditary with respect to restriction. So, the most basic question in this context is how to characterise those $\mathcal{Y} \subseteq \mathcal{P}(X)$ and $i \in Inf(X)$ for which this obviously sufficient condition is also necessary, that is, for which

$$\{i_X \in Inf(X) \mid (i, i_X) \in Cpb(X)\}$$

equals the set

$$\{i_X \in Inf(X) \mid (i|_Y, i_X|_Y) \in Cpb(Y) \text{ for all } Y \in \mathcal{Y}\}.$$

- given $\mathcal{Y} \subseteq \mathcal{P}(X)$ and a family of elements $i_Y \in Inf(Y)$ $(Y \in \mathcal{Y})$, when does there exist some - or even a unique - element $i_X \in Inf(X)$ with $i_X|_Y = i_Y$ for all $Y \in \mathcal{Y}$?

In the following, we'll first collect some of what is already known regarding these questions for special choices of $\mathcal{Y} \subseteq \mathcal{P}(X)$ and of the family $(i_Y)_{Y\in\mathcal{Y}}$, and then we'll address question (3) in some more detail allowing $\mathcal{Y}$ and $(i_Y)_{Y\in\mathcal{Y}}$ to be almost arbitrary.

2. Some Previous Results

2.1 Hierarchies

According to well-established traditions (cf. [G87]), a *hierarchy* $\mathcal{C}$ defined on a set X - or, for short, an $X-$*hierarchy* - is defined to be a subset $\mathcal{C}$ of $\mathcal{P}(X)$ such that

$$C_1 \cap C_2 \in \{C_1, C_2, \emptyset\}$$

holds for all $C_1, C_2 \in \mathcal{C}$. For technical reasons, we require also that $\emptyset \in \mathcal{C}$ and $X \in \mathcal{C}$ should hold for any $X-$hierarchy $\mathcal{C}$ which then automatically implies also that $\mathcal{C}$ is closed with respect to intersection. Given a similarity measure s defined on X (that is, just a map $s : X \times X \to \mathbb{R}$ satisfying the symmetry condition $s(x, y) = s(y, x)$ for all $x, y \in X$), an $X-$hierarchy $\mathcal{C}_s$ can be associated with s according to the definition

$$\mathcal{C}_s := \{C \subseteq X \mid s(a, b) > s(a, c) \text{ for all } a, b \in C \text{ and } c \in X - C\}.$$

It is well known that, for any $X-$hierarchy $\mathcal{C} \subseteq \mathcal{P}(X)$, one can always find some similarity measure s defined on X with $\mathcal{C} = \mathcal{C}_s$ and that, with $n := \#X$, one always has $\#\mathcal{C} \leq 2n$. This can either be seen by induction, using the fact that for every maximal cluster $C_0 \in \mathcal{C}$ which is different from X, $\mathcal{C}$ decomposes into the three sets $\{X\}$, $\{C \in \mathcal{C} \mid C \subseteq C_0\}$ and $\{C \in \mathcal{C} \mid C \cap C_0 = \emptyset\}$ - with the last two having only the empty set $\emptyset \in \mathcal{P}(X)$ in common. Following an idea of Boris Mirkin (cf. [M97]), it can also be deduced as follows: Given an $X-$hierarchy $\mathcal{C}$ and a cluster $C \in \mathcal{C}$, let

$$V(C) \subseteq I_{\mathbb{R}}(X) := \{f : X \to \mathbb{R} \mid \sum_{x\in X} f(x) = 0\}$$

denote the real vector space of all maps f from X into $\mathbb{R}$ which vanish outside C, are orthogonal to every constant map (relative to the canonical inner product defined on $\mathbb{R}^X$), and are constant on every proper subcluster C' of C. It is easily seen that $V(C)$ is different from 0 if and only if C contains at least two distinct elements, and that any two such subspaces $V(C)$ and $V(C')$ are orthogonal to each other for any $C, C' \in \mathcal{C}$ with $C \neq C'$. Actually, a simple induction argument similar to the one used just above establishes that $I_{\mathbb{R}}(X)$ is the orthogonal direct sum of the spaces $V(C)$ $(C \in \mathcal{C})$. Yet, even without establishing this fact, the construction yields, for $n := \#X$ as above, the inequality

$$\#\mathcal{C} \leq \#\{C \in \mathcal{C} \mid \#C \leq 1\} + \dim_{\mathbb{R}} I_{\mathbb{R}}(X) \leq (1 + n) + (n - 1) = 2n$$

- and, hence, shows also that for an $X-$hierarchy $\mathcal{C} \subseteq \mathcal{P}(X)$ with $\#\mathcal{C} = 2n$ one has necessarily $\dim_{\mathbb{R}} V(C) = 1$ for each $C \in \mathcal{C}$ with $\#C > 1$.

All that can be put easily into the framework considered in the previous section: First, we note that a set system $\mathcal{C} \subseteq \mathcal{P}(X)$ with $\emptyset \in \mathcal{C}$ and $X \in \mathcal{C}$ is an $X-$hierarchy if and only if $\mathcal{C}|_Y = \{C \cap Y \mid C \in \mathcal{C}\} \subseteq \mathcal{P}(Y)$ is a $Y-$hierarchy for every $Y \subseteq X$ with $\#Y = 3$: indeed, if there would exist some subsets $C_1, C_2 \in \mathcal{C}$ with $\emptyset \neq C_1 \cap C_2 \neq C_i$ for $i = 1, 2$, then putting $Y := \{a, b, c\}$ with $a \in C_1 \cap C_2$, $b \in C_1 - (C_1 \cap C_2)$ and $c \in C_2 - (C_1 \cap C_2)$ produces a subset Y of cardinality 3 so that $\mathcal{C}|_Y$ is not a $Y-$hierarchy. Hence, putting

$$\mathcal{Y} := \mathcal{P}_{\leq 3}(X) = \{Y \subseteq X \mid \#Y \leq 3\}$$

and choosing, for each $Y \in \mathcal{Y}$, a $Y-$hierarchy $i_Y \in \mathcal{P}(\mathcal{P}(Y)) = Cl^{aff}(Y)$, the resulting set system

$$\bar{i}_X := \{C \subseteq X \mid C \cap Y \in i_Y \text{ for all } Y \in \mathcal{Y}\}$$

surely is an $X-$hierarchy. In addition, given either an $X-$hierarchy $\mathcal{C} \subseteq \mathcal{P}(X)$ or a similarity measure s defined on X, we may consider the $Y-$hierarchies

$$i_Y = i_Y^{\mathcal{C}} := \mathcal{C}|_Y = \{C \cap Y \mid C \in \mathcal{C}\} \in Cl^{aff}(Y)$$

or

$$i_Y = i_Y^s := \mathcal{C}_{s|_{Y \times Y}} \in Cl^{aff}(Y),$$

respectively, for every $Y \in \mathcal{Y}$. It is then easy to see that $\mathcal{C}$ (or $\mathcal{C}_s$) coincides with

$$\bar{i}_X := \{C \subseteq X \mid C \cap Y \in i_Y \text{ for all } Y \in \mathcal{Y}\} \in Cl^{aff}(X).$$

This holds essentially by definition in case we start with a similarity measure s because, given some $C \subseteq X$, we clearly have $s(a, b) > s(a, c)$ for all $a, b \in C$ and $c \in X - C$ if and only if, for any $Y \in \mathcal{Y}$, the same inequality $s(a, b) > s(a, c)$ holds for all $a, b \in C \cap Y$ and $c \in Y \backslash C$, that is, if and only if $C \cap Y \in \mathcal{C}_{s|_{Y \times Y}}$ holds. If, on the other hand, we start with an $X-$hierarchy $\mathcal{C}$, then clearly $\mathcal{C} \subseteq \bar{i}_X$ holds by definition of $\bar{i}_X$, while $\bar{i}_X \subseteq \mathcal{C}$ - and, hence, equality - holds for the following reasons: first, given some $C \in \bar{i}_X$, we may assume w.l.o.g that $\emptyset \neq C \neq X$ holds, so we can find elements $a \in C$ and $b \in X - C$. Next, for any $x \in C$, we can find some $C' = C'_x \in \mathcal{C}$ with

$$\{a, x\} = C \cap \{a, b, x\} = C'_x \cap \{a, b, x\},$$

that is, with $a, x \in C'_x$ and $b \notin C'_x$. As $\mathcal{C}$ is a hierarchy and we have $a \in C'_x \cap C'_y$ for all $x, y \in C$, we must have $C'_x \subseteq C'_y$ or $C'_y \subseteq C'_x$ for all $x, y \in C$, so the set of these clusters is linearly ordered and, hence, contains a unique largest cluster $C' = C'(b) \in \mathcal{C}$ which still does not contain b, but - in view of $x \in C'_x \subseteq C'(b)$ - contains every element x from C; so, for every $b \in X - C$, we can find some cluster $C'(b) \in \mathcal{C}$ with

$$C \subseteq C'(b) \subseteq X - \{b\}.$$

Hence, we get

$$C = \bigcap_{b \in X - C} C'(b) \in \mathcal{C},$$

as claimed.

Finally, it can also be shown that whenever a $Y-$hierarchy $i_Y \subseteq \mathcal{P}(Y)$ is given for every $Y \in \mathcal{Y}$, the corresponding $X-$hierarchy

$$\bar{i}_X := \{C \subseteq X \mid C \cap Y \in i_Y \text{ for all } Y \in \mathcal{Y}\}$$

satisfies the relation

$$\bar{i}_X|_Y = i_Y$$

for every $Y \in \mathcal{Y}$ if and only if that holds for every subset $Z \subseteq X$ of cardinality at most 4 and the correspondingly defined $Z-$hierarchy

$$\bar{i}_Z := \{C \subseteq Z \mid C \cap Y \in i_Y \text{ for every } Y \in \mathcal{Y} \cap \mathcal{P}(Z)\}$$

(and every $Y \in \mathcal{Y} \cap \mathcal{P}(Z)$): It is obvious that this condition is necessary in view of $\bar{i}_X|_Z \subseteq \bar{i}_Z$ and, hence, $i_Y = \bar{i}_X|_Y = \bar{i}_X|_Z|_Y \subseteq \bar{i}_{Z|Y} \subseteq i_Y$.

To establish the converse, one may define

$$\langle a, b\rangle := \{c \in X \mid c \in \{a, b\} \text{ or } \{a, b\} \notin i_{\{a,b,c\}}\}$$

for all $a, b \in X$ (whether $a \neq b$ or $a = b$) and first note that $Y \in \mathcal{Y}$, $i_Y|_{Y'} \subseteq i_{Y'}$, for all $Y' \subseteq Y$, and $\langle a, b\rangle \in \bar{i}_X$ for all $a, b \in Y$ will imply $\bar{i}_X|_Y = i_Y$ because we surely have $\emptyset, Y \in \bar{i}_X|_Y$, while every other subset in i_Y is of the form $\{a, b\}$ for some $a, b \in Y$ in which case we have $\{a, b\} = \langle a, b\rangle \cap Y \in \bar{i}_X|_Y$ because $c \in Y - \{a, b\}$ implies $c \notin \langle a, b\rangle$ in view of $\{a, b\} \in i_Y$ and, therefore, $\{a, b\} \in i_Y|_{\{a,b,c\}} \subseteq i_{\{a,b,c\}}$.

Next, note that - vice versa - $\bar{i}_X|_Y = i_Y$ for all $Y \in \mathcal{Y}$ also implies $\langle a, b\rangle \in \bar{i}_X$ for all $a, b \in X$ because the smallest subset C in $\bar{i}_X$ containing a and b (that is, the intersection of all those subsets) surely must contain $\langle a, b\rangle$ in view of $\{a, b\} \subseteq C \cap \{a, b, c\} \in i_{\{a,b,c\}}$ for all $c \in X$ and, therefore, $c \in C$ for all $c \in X$ with $\{a, b\} \notin i_{\{a,b,c\}}$, while it cannot contain any c not in $\langle a, b\rangle$ because for any such c we have $c \notin \{a, b\}$ and $\{a, b\} \in i_{\{a,b,c\}} = \bar{i}_X|_{\{a,b,c\}}$, so there must exist some $C' \in \bar{i}_X$ with $C' \cap \{a, b, c\} = \{a, b\}$ which surely implies $C \subseteq C'$ as well as $c \notin C'$ and, hence, $c \notin C$, as claimed.

So, it remains to show that we have $\langle a, b\rangle \in \bar{i}_X$ for all $a, b \in X$ whenever we have $\langle a, b\rangle \cap Z \in \bar{i}_Z$ for all $Z \subseteq X$ with $a, b \in Z$ and $\#Z \leq 4$ or - equivalently - whenever we have $\langle a, b\rangle \cap \{b, c, d\} \in i_{\{b,c,d\}}$ for all $a, b, c, d \in X$. So, assume the latter, pick $a, b \in X$ and consider $\langle a, b\rangle \cap Y$ for some $Y \in \mathcal{Y}$. If $\#(\{a, b\} \cup Y) \leq 4$, then $\langle a, b\rangle \cap (\{a, b\} \cup Y) \in \bar{i}_{\{a,b\}\cup Y}$ by assumption, so we also clearly have

$$\langle a, b\rangle \cap Y = (\langle a, b\rangle \cap (\{a, b\} \cup Y)) \cap Y \in \bar{i}_{\{a,b\}\cup Y|Y} \subseteq i_Y.$$

Otherwise, we have $a \neq b$, $\#Y = 3$ and $a, b \notin Y$. Assume $\langle a, b\rangle \cap Y \notin i_Y$. Then there exist $x, y, z \in X$ with $x \in \langle a, b\rangle \cap Y$, $y \notin \langle a, b\rangle \cap Y$ and $Y = \{x, y, z\}$. From our assumption, we get $\{a, x\} = \langle a, b\rangle \cap \{a, x, y\} \in i_{\{a,x,y\}}$, that is $y \notin \langle a, x\rangle$. Hence, if also $z \notin \langle a, b\rangle \cap Y$, we can replace y by z in the above argument which leads to $z \notin \langle a, x\rangle$ and, hence,

$$\langle a, b\rangle \cap \{x, y, z\} = \{x\} = \langle a, x\rangle \cap \{x, y, z\} \in i_{\{x,y,z\}} = i_Y,$$

while, if $z \in \langle a, b\rangle \cap Y$, we can replace x by z in the above argument which leads to $y \notin \langle a, z\rangle$. Hence, as $z \in \langle a, x\rangle$ or $x \in \langle a, z\rangle$ because we cannot simultaneously have $\{a, x\} \in i_{\{a,x,z\}}$ **and** $\{a, z\} \in i_{\{a,x,z\}}$, we have either

$$\langle a, b\rangle \cap Y = \{x, z\} = \langle a, x\rangle \cap Y$$

or we have

$$\langle a, b\rangle \cap Y = \{x, z\} = \langle a, z\rangle \cap Y,$$

so - in any case - we have $\langle a, b\rangle \cap Y \in i_Y$, as claimed.

Another way to deduce the same result is also of some interest: given a family of hierarchies $i_Y (Y \in \mathcal{Y})$, define a ternary relation $ab|c$ on X by

$$ab|c \Leftrightarrow c \neq a, b \text{ and } \{a, b\} \in i_{\{a,b,c\}}$$

$(a, b, c \in X)$. Clearly, one has
(H1) $ab|c \quad \Leftrightarrow \quad ba|c$
and
(H2) $ab|c \;\&\; ac|d \Rightarrow b \neq d$
for all $a, b, c, d \in X$, the latter because $b = d$ would imply $\#\{a, b, c\} = 3$ as well as well as $\{a, b\} \in i_{\{a,b,c\}}$ and $\{a, c\} \in i_{\{a,b,c\}}$ in contradiction to $\emptyset \neq \{a\} = \{a, b\} \cap \{a, c\} \neq \{a, b\}, \{a, c\}$. And it is also clear that for any ternary relation satisfying (H1) and (H2) the set system $\mathcal{C}$ consisting of all subsets $C \subseteq X$ with $ab|c$ for all $a, b \in C$ and $c \in X - C$ is a hierarchy.

Next observe that in case

$$\langle a, b\rangle \cap \{b, c, d\} \in i_{\{b,c,d\}}$$

holds for all $a, b, c, d \in X$, one also has
(H3) $ab|c \;\&\; ac|d \Rightarrow ab|d$
as well as
(H4) $ab|d \;\&\; ac|d \Rightarrow bc|d$
for all $a, b, c, d \in X$. One can then show that there is a canonical 1-1 correspondence between (a) ternary relations defined on a finite set X satisfying (H1), (H2), (H3) and (H4) and (b) hierarchies $\mathcal{C} \subseteq \mathcal{P}(X)$, given by associating to any such relation - as above - the set system $\mathcal{C}$ consisting of all subsets $C \subseteq X$ with $ab|c$ for all $a, b \in C$ and $c \in X - C$, or - vice versa - to any $X-$hierarchy $\mathcal{C}$ the ternary relation defined by

$$ab|c \Leftrightarrow \text{ there exists some } C \in \mathcal{C} \text{ with } a, b \in C \text{ and } c \in X - C.$$

The fact that this sets up indeed a one-to-one correspondence is almost equivalent to the above-mentioned fact that, for any family $i_Y (Y \in \mathcal{Y})$ of $Y-$hierarchies, we have $\bar{i}_X|_Y = i_Y$ for all $Y \in \mathcal{Y}$ if and only we have $\bar{i}_Z|_Y = i_Y$ for all $Z \subseteq X$ with $\#Z \leq 4$ and all $Y \in \mathcal{Y} \cap \mathcal{P}(Z)$, so any one of these two facts can be deduced easily from the other one.

Altogether, we see that the theory of hierarchies - including the way they are related to similarity measures - fits perfectly well into the conceptual framework developed in the previous section.

2.2 Trees

Given a set X, a tree structure on X is a triple (V, E, ϕ) consisting of a set V - called the set of *vertices* or *nodes* of that tree structure - , a subset E of the set $\mathcal{P}_2(V) := \{e \subseteq V \mid \#e = 2\}$ of subsets of V of cardinality 2 - called the set of edges -, and a map $\phi : X \to V$ such that the pair (V, E) is a tree, that is, such that for any two vertices v, v' there exists one and only one pair $(d, p) = (d(v, v'), p(v, v'))$ consisting of a natural number

$$d = d(v, v') = d^E(v, v') \in \mathbb{N}_0$$

- called the (combinatorial or unweighted) *distance* between v and v' (relative to (V,E)) - and a sequence

$$p = p(v,v') = p^E(v,v') = (v_0, v_1, \dots, v_d)$$

of length $d+1$ of vertices

$$v_i = p_i(v,v') = p_i^E(v,v')$$

$(i = 0, \dots, d)$ from V with $v_0 = v, v_d = v'$, and with

$$e_i = e_i(v,v') = e_i^E(v,v') := \{v_{i-i}, v_i\} \in E$$

for all $i = 1, \dots, d$ as well as $v_{i-1} \neq v_{i+1}$ for all $i = 1, \dots, d-1$, called the *path* from v to v' (in the tree (V,E)) (cf. [BD86]). It is easily seen that - with these notations - one must have $\#\{v_0, v_1, \dots, v_d\} = d+1$ and $\#\{e_1, \dots, e_d\} = d$, that the set

$$\Delta(v,v') = \Delta_E(v,v') := \{e_1, \dots, e_d\}$$

coincides with the intersection of all subsets E' of E for which there exist vertices $w_0 := v, w_1, \dots, w_k := v'$ from V with $\{w_{i-1}, w_i\} \in E'$ for all $i = 1, \dots, k$, and that for $v, v', v'' \in V$ one has $p_i(v,v') = p_i(v,v'')$ for some i (assumed to be at most equal to $min(d(v,v'), d(v,v''))$ so that both terms are defined) if and only if i is smaller than or equal to

$$d(v; v', v'') := \#(\Delta(v,v') \cap \Delta(v,v''))$$

and, hence,

$$\Delta(v',v'') = (\Delta(v,v') \cup \Delta(v,v'')) - (\Delta(v,v') \cap \Delta(v,v'')),$$

the symmetric difference between $\Delta(v,v')$ and $\Delta(v,v'')$.

A tree structure (V,E,ϕ) defined on X is said to be an $X-$tree if (i) E coincides with its subset $\bigcup_{x,y \in X} \Delta(\phi(x), \phi(y))$ and if (ii) for every vertex $v \in V - \phi(X)$ there exist at least three distinct edges $e_1, e_2, e_3 \in E$ containing v. Equivalently, defining an equivalence relation "$\overset{e}{\sim}$" on X for each $e \in E$ by

$$x \overset{e}{\sim} y \Leftrightarrow e \notin \Delta(\phi(x), \phi(y)),$$

a tree structure (V,E,ϕ) defined on X is an $X-$tree if (i) none of these equivalence relations is trivial in which case the associated set of equivalence classes consists of exactly two (non-empty!) subsets of X and, hence, constitutes in particular a split $S_e \in \mathcal{S}p(X)$, and if (ii) for any two edges $e, e' \in E$ one has

$$S_e = S_{e'} \Leftrightarrow e = e'.$$

It is easily seen and well known that by associating to any tree structure (V,E,ϕ) defined on a set X the triple (V',E',ϕ') defined by

$$\begin{aligned} V' &:= \phi(X) \cup \{v \in V \mid 3 \leq \#\{S_e \mid v \in e \in E, \#S_e = 2\}\}, \\ E' &:= \{\{v,w\} \in \mathcal{P}_2(V') \mid \text{there exist } e, f \in E \\ &\qquad \text{with } v \in e, w \in f \text{ and } S_e = S_f \neq \{X\}\}, \end{aligned}$$

and

$$\phi' : X \to V' : x \mapsto \phi(x)$$

is an $X-$tree which is called the $X-$tree induced by the tree structure (V,E,ϕ). Clearly $(V,E,\phi) = (V',E',\phi')$ if and only if (V,E,ϕ) is an $X-$tree.

It is also easily seen and well-known (see for instance [B71] and [BD86]) that, for any two tree structures (V_1, E_1, ϕ_1) and (V_2, E_2, ϕ_2) defined on X, the following assertions are equivalent:

(A1) *The set* $\mathcal{S}(V_1, E_1, \phi_1) := \{S_{e_1}, \mid e_1 \in E_1, \#S_{e_1} = 2\}$ *coincides with the correspondingly defined set* $\mathcal{S}(V_2, E_2, \phi_2)$.

(A2) *There exist maps* $w_1 : E_1 \to \mathbb{R}_{>0}$ *and* $w_2 : E_2 \to \mathbb{R}_{>0}$ *such that for all* $x, y \in X$ *and* $\Delta_1 := \Delta_{E_1}(\phi_1(x), \phi_1(y))$, $\Delta_2 := \Delta_{E_2}(\phi_2(x), \phi_2(y))$ *one has*

$$\sum_{e_1 \in \Delta_1} w_1(e_1) = \sum_{e_2 \in \Delta_2} w_2(e_2).$$

(A3) *For the corresponding induced* $X-$*trees* (V_1', E_1', ϕ_1') *and* (V_2', E_2', ϕ_2'), *one has*

$$d^{E_1'}(\phi_1'(x), \phi_1'(y)) = d^{E_2'}(\phi_2'(x), \phi_2'(y))$$

for all $x, y \in X$.

(A4) *There exists one (and only one!) bijection* $\alpha : V_1' \xrightarrow{\sim} V_2'$ *between the sets of vertices of the corresponding induced* $X-$*trees such that* $E_2' = \{\alpha(e_1') \mid e_1' \in E_1'\}$ *and* $\phi_2' = \alpha \circ \phi_1'$.

If one and, hence, all of these assertions hold, the tree structures (V_1, E_1, ϕ_1) and (V_2, E_2, ϕ_2) are called *equivalent.*

Clearly, any tree structure (V, E, ϕ) defined on X is equivalent to its induced $X-$tree and that $X-$tree is determined uniquely up to canonical isomorphism by the equivalence class of (V, E, ϕ). It is also well-known that for any $X-$tree (V, E, ϕ) and any two edges $e, e' \in E$, the two associated splits S_e and $S_e' \in \mathcal{S}_p(X)$ are *compatible*, that is, the set $\{A \cap A' \mid A \in S_e, A' \in S_{e'}\}$ contains at most three non-empty subsets of X. And it has been established by P. Buneman (cf. [B71]) that, vice versa, for a finite set X and any set $\mathcal{S} \subseteq \mathcal{S}p(X)$ of splits $S = \{A, B\} \in \mathcal{S}p(X)$ which are pairwise compatible, that is, with

$$\#\{A \cap A' \mid A \in S, A' \in S', A \cap A' \neq \emptyset\} \leq 3$$

for all $S, S' \in \mathcal{S}$, there exists a tree structure (V, E, ϕ) for X with $\mathcal{S}(V, E, \phi) = \mathcal{S}$.

Actually, assuming without loss of generality $\{X, \emptyset\} \notin \mathcal{S}$, an $X-$tree (V, E, ϕ) with that property can be defined as follows: put

$$\begin{aligned} V &:= \{v : \mathcal{S} \to \mathcal{P}(X) \mid v(S) \in S \text{ and } v(S_1) \cap v(S_2) \neq \emptyset \text{ for all } S, S_1, S_2 \in \mathcal{S}\}, \\ E &:= \{\{v, w\} \in V \mid \#\{S \in \mathcal{S} \mid v(S) \neq w(S)\} = 1\}, \end{aligned}$$

and

$$\phi : X \to V : x \mapsto (v_x : \mathcal{S} \to \mathcal{P}(X) : S \mapsto S(x)),$$

where $S(x) \in S$ denotes that subset A of X contained - as an element - in S which contains x.

Hence, (equivalence classes of) tree structures defined on a finite set X can be identified with subsets $\mathcal{S}$ of the set $\mathcal{S}p(X)$ of all splits of X in which any two splits $S, S' \in \mathcal{S}$ are *compatible*, that is, satisfy the above condition

$$\#\{A \cap A' \mid A \in S, A' \in S', A \cap A' \neq \emptyset\} \leq 3,$$

and none coincides with $\{X, \emptyset\}$.

Consequently, also the theory of tree structures fits nicely into the framework proposed in the previous section: given a finite set X and a subset $\mathcal{S}$ of the set $\mathcal{S}p(X)$ of all splits of X, it is easy to see that $\mathcal{S}$ represents a tree structure of X - that is, one has

$$\#\{A \cap A' \mid A \in S, A' \in S', A \cap A' \neq \emptyset\} \leq 3$$

for all $S, S' \in \mathcal{S}$ - if and only if this holds for $\mathcal{S}|_Y = \{\{A \cap Y, B \cap Y\} \mid \{A, B\} \in \mathcal{S}\}$ for all $Y \subseteq X$ with $\#Y \leq 4$.

Vice versa, putting $\mathcal{Y} := \mathcal{P}_{\leq 4}(X) := \{Y \subseteq X \mid \#Y \leq 4\}$ and choosing, for each $Y \in \mathcal{Y}$, a set $i_Y \subseteq \mathcal{S}p(Y)$ of pairwise compatible splits - including, for the sake of technical simplicity - the trivial split $\{Y, \emptyset\}$ induced by the trivial equivalence relation, the associated element $\bar{i}_X \in Cl^{proj}(X)$ defined by

$$\bar{i}_X := \{\{A, B\} \in \mathcal{S}p(X) \mid \{A \cap Y, B \cap Y\} \in i_Y \text{ for all } Y \in \mathcal{Y}\}$$

always represents a tree structure on X.

It is also easy to see that in case $\#X = n + 1$, one has $\#\mathcal{S} \leq 2n$ for all subsets $\mathcal{S} \subseteq \mathcal{S}p(X)$ of pairwise compatible splits (including possibly the empty split), either by proving this directly or by - after choosing some $x_0 \in X$ to play the rôle of the "point at infinity" - replacing $\mathcal{S} \subseteq \mathcal{S}p(X)$ by

$$\mathcal{C}^{x_0}(\mathcal{S}) := \{A \subseteq X - \{x_0\} \mid \{A, X - A\} \in \mathcal{S}\}$$

and noting that $\mathcal{C}^{x_0}(\mathcal{S}) \cup \{\emptyset, X - \{x_0\}\}$ is necessarily an $(X - \{x_0\})$ - hierarchy whenever any two splits in $\mathcal{S}$ are compatible - actually, any two splits in $\mathcal{S}$ are pairwise compatible if and only if $\mathcal{C}^x(\mathcal{S}) \cup \{\emptyset, X - \{x\}\}$ is an $(X - \{x\})$ - hierarchy for every $x \in X$.

To construct an $X-$tree from local data in this way, the decision about which tree structure to choose for any given small subset $Y \in \mathcal{Y}$ can be based on whatever creed one adheres to: given a distance $d : X \times X \to \mathbb{R}$ defined on X, one might - for any $Y \in \mathcal{Y}$ - take all splits $\{A, B\}$ of Y with

$$d(a, a') + d(b, b') < d(a, b) + d(a', b')$$

for all $a, a' \in A$ and $b, b' \in B$ in which case the corresponding family $\bar{i}_X$ of splits of X is exactly the set $\mathcal{S}_{Buneman}(d)$ of all splits $\{A, B\}$ of X satisfying exactly the same requirement. As indicated by our notation, this set of splits and the corresponding tree structure was considered already by P. Buneman in the paper [B71] mentioned already above which, by the way, was devoted to archeological classification.

If X is a set of sequences $x = (x_1, \ldots, x_k)$ whose entries x_i all come from some alphabet $\mathcal{A}$ on which a metric $d_{\mathcal{A}}$ is defined (e.g. the trivial metric $d_{\mathcal{A}}(a, b) := 1 - \delta_{ab}$), there are many alternatives to define $i_Y \in \mathcal{S}p(Y)$ for all $Y \in \mathcal{Y}$: If all sequences are of the same length k, one can define a metric d on X by

$$d(x, y) := \sum_{i=1}^{k} d_{\mathcal{A}}(x_i, y_i)$$

for all $x = (x_1, \ldots, x_k), y = (y_1, \ldots, y_k) \in X$ and then proceed as above. One may also explore, for each $Y \in \mathcal{Y}$, the very few possible tree structures definable on Y and then decide for the most parsimonious tree structure as the appropriate local input. In general, one will first have to align the sequences according to some alignment score, and one might then use just that score (or a distance measure derived from it) for constructing the Buneman splits (to which end one may actually restrict oneself to pairwise alignment). Instead, one might try to simultaneously construct, for each $Y \in \mathcal{Y}$, the tree structure as well as the alignment - again, say, by exploring all possible tree structures - and then invoke a parsimony or a maximum likelihood principle.

For real data sets, it might actually be advisable to explore all or, at least, quite a few of these alternatives as only those splits can be trusted as being reliable which are observed in many of the resulting $X-$trees; - actually, an extensive literature exists regarding how to construct a *consensus* tree structure from many given ones (cf. for example [DM85]) which could also be evoked at that stage, even though in most practical cases - at least, when it comes to problems in phylogeny - there will be only a few doubtful splits which should rather be discussed individually, taking into account all sorts of arguments and not exclusively only those which are based on formal tree-construction and/or consensus procedures.

To conclude this subsection, we just mention that in analogy to the affine case, given a family $i_Y \in Cl^{proj}(Y)$ $(Y \in \mathcal{Y} = \mathcal{P}_{\leq 4}(X))$ of tree structures, there exists a tree structure $i_X \in Cl^{proj}(X)$ with $i_X|_Y = i_Y$ for all $Y \in \mathcal{Y}$ if and only if this holds for any subset Z of X of cardinality at most 5, that is, if and only if for any subset Z of X with $\#Z \leq 5$ there exists a tree structure $i_Z \in Cl^{proj}(Z)$ with $i_Z|_Y = i_Y$ for all $Y \in \mathcal{Y} \cap \mathcal{P}(Z)$ (cf. [BD86]).

2.3 Weak Hierarchies and Weakly Compatible Split Systems

Next, I want to point out that also the theory of weak hierarchies and weakly compatible split systems as developed in [BD89] and [BD92] fits nicely into the above framework. According to [BD89], a *weak hierarchy* $\mathcal{C}$ defined on a set X is a subset of $\mathcal{P}(X)$ such that $C_1 \cap C_2 \cap C_3 \in \{C_1 \cap C_2,\ C_2 \cap C_3,\ C_3 \cap C_1\}$ holds for all $C_1, C_2, C_3 \in \mathcal{C}$.

It follows that a subset $\mathcal{C}$ of $\mathcal{P}(X)$ is a weak hierarchy defined on X if and only if $\mathcal{C}|_Y := \{C \cap Y \,|\, C \in \mathcal{C}\}$ is a weak hierarchy defined on Y for every $Y \in \mathcal{Y} := \mathcal{P}_{\leq 3}(X)$ which in turn is the case if and only if $\mathcal{C}|_Y$ does not contain all three subsets of Y of cardinality 2 for every $Y \subseteq X$ of cardinality 3. Hence, given a weak hierarchy $i_Y \subseteq \mathcal{P}(Y)$ for every $Y \in \mathcal{Y}$, the corresponding subset $\bar{i}_X = \{C \subseteq X \,|\, C \cap Y \in i_Y \text{ for } Y \in \mathcal{Y}\} \subseteq \mathcal{P}(X)$ of $\mathcal{P}(X)$ will always be a weak hierarchy, too. Moreover, it follows easily from adapting the first (rather trivial) part of an argument presented in **2.1** to this situation, that now we have $res_{X \to Y}(\bar{i}_X) = i_Y$ for all $Y \in \mathcal{Y}$ in case (i) every weak hierarchy i_Y contains Y and the empty set among its clusters and is closed with respect to intersection and (ii) we have $res_{Z \to Y}(\bar{i}_Z) = i_Y$ for all $Z \subseteq X$ of cardinality at most 5 and all $Y \in \mathcal{Y} \cap \mathcal{P}(Z)$ (with $\bar{i}_Z := \{C \subseteq Z \,|\, C \cap Y \in i_Y \text{ for all } Y \in \mathcal{Y} \cap \mathcal{P}(Z)\}$, of course).

That it is not enough to require condition (ii) for all subsets $Z \subseteq X$ of cardinality at most 4 can be deduced from the following simple (counter)example: put $X := \{1,2,3,4,5\}$, put $i_Y := \mathcal{P}(Y)$ if $\#Y \leq 2$, and in case $\#Y = 3$ put

$$i_Y := \begin{cases} \mathcal{P}(Y)\backslash\{\{1,2\},\{3,4\}\} & \text{if } 5 \notin Y, \\ \{Y\} \cup \mathcal{P}_{\leq 1}(Y) & \text{if } Y = \{3,4,5\}, \\ \mathcal{P}(Y) - \{\{a,5\} \mid a \in Y - \{5\}\} & \text{else.} \end{cases}$$

It is easy to see that this implies

$$\begin{aligned} \bar{i}_{\{1,2,3,4\}} &= \{\{1,2,3,4\}\} \cup \mathcal{P}_{\leq 2}(\{1,2,3,4\}) - \{\{1,2\},\{3,4\}\}, \\ \bar{i}_{\{1,2,a,5\}} &= \{\{1,2,a,5\},\{1,2,a\},\{1,a\},\{2,a\}\} \cup \mathcal{P}_{\leq 1}(\{1,2,a,5\}) \end{aligned}$$

for $a = 3,4$ and

$$\bar{i}_{\{a,3,4,5\}} = \{\{a,3,4,5\},\{a,3\},\{a,4\}\} \cup \mathcal{P}_{\leq 1}(\{a,3,4,5\})$$

for $a = 1,2$ from which formulae one can easily conclude that condition (ii) is fulfilled in this example for all $Z \subseteq X$ with $\#Z \leq 4$. Yet, there can't be a subset $C \subseteq X$ with $C \in \bar{i}_X$ (that is, with $C \cap Y \in i_Y$ for all $Y \subseteq X$ with $\#Y \leq 3$) as well as $C \cap \{1,2,5\} = \{1,2\}$ because any $C \in \bar{i}_X$ with $\{1,2\} \subseteq C$ must intersect $\{1,2,a\}$ ($a \in \{3,4\}$) in the only subset in $i_{\{1,2,a\}}$ containing $\{1,2\}$ which is $\{1,2,a\}$ itself, so we must have $3,4 \in C$ which implies that C must intersect $\{3,4,5\}$ in the only subset in $i_{\{3,4,5\}}$ containing $\{3,4\}$ which is $\{3,4,5\}$ itself. So, we must have $5 \in C$ and, hence, $C \cap \{1,2,5\} \neq \{1,2\}$, as claimed.

It is easy to see that one has $\#\mathcal{C} \leq \#\mathcal{P}_{\leq 2}(X) = \binom{n}{2} + \binom{n}{1} + \binom{n}{0}$ for every weak hierarchy defined on a set X of cardinality n: Indeed this follows easily from the fact that for any non-empty cluster C in a weak hierarchy $\mathcal{C}$ there exist $a, b \in C$ with $C \subseteq C'$ for all $C' \in \mathcal{C}$ with $a, b \in C'$ because otherwise there would exist a smallest subset T of C of cardinality > 2 and $C \subseteq C'$ for every $C' \in \mathcal{C}$ with $T \subseteq C'$, so for any three distinct elements $a_1, a_2, a_3 \in T$ there would exist, for each $i \in \{1,2,3\}$, some cluster $C_i \in \mathcal{C}$ with $a_i \notin C_i$ and $T - \{a_i\} \subseteq C_i$, in contradiction to $C_1 \cap C_2 \cap C_3 \in \{C_1 \cap C_2,\ C_2 \cap C_3,\ C_3 \cap C_1\}$ for all $C_1, C_2, C_3 \in \mathcal{C}$.

Next, it is obvious that for any weak hierarchy $\mathcal{C} \subseteq \mathcal{P}(X)$ we have $\mathcal{C} \subseteq \bar{i}_X$ for the weak hierarchy $\bar{i}_X$ associated with the family $i_Y := \mathcal{C}|_Y$ ($Y \in \mathcal{Y}$), and that equality implies that $\mathcal{C}$ is closed with respect to intersection provided that that holds for all i_Y ($Y \in \mathcal{Y}$). More precisely (see below), it can be shown that $\bar{i}_X$ is always contained in the smallest intersection-closed subset $\hat{\mathcal{C}}$ of $\mathcal{P}(X)$ containing $\{X\} \cup \mathcal{C}$ (which - for a weak hierarchy $\mathcal{C}$ - is easily seen to coincide with $\{X\} \cup \{C_1 \cap C_2 \mid C_1, C_2 \in \mathcal{C}\}$) and, hence, that $\hat{\mathcal{C}}$ always coincides with the weak hierarchy $\bar{j}_X$ associated with the family $j_Y := \hat{i}_Y$ (with $\hat{i}_Y$, of course, denoting the smallest intersection-closed subset of $\mathcal{P}(Y)$ containing $\{Y\} \cup i_Y$). Note also that i_Y will always coincide with $\hat{i}_Y = j_Y$ provided $\mathcal{C}$ contains X as well as every subset $C \subseteq X$ of cardinality ≤ 1; so in this case, we will always have $\bar{i}_X = \hat{\mathcal{C}}$.

Finally, given a symmetric map $s : X^2 \to \mathbb{R}$, the set system $\mathcal{H}_s$ defined as

$$\mathcal{H}_s(C) := \{C \subseteq X \mid s(a,b) > min(s(a,c), s(b,c)) \text{ for all } a, b \in C \text{ and } c \in X - C\}$$

always is an intersection-closed weak hierarchy (which was the starting point for their discussion in [BD89]). Clearly, $\mathcal{H}_s$ is the weak hierarchy $\bar{i}_X$ associated with

the family of weak hierarchies $i_Y := \mathcal{H}_s(C|_{Y\times Y})$ $(Y \in \mathcal{Y})$. So, as in the case of hierarchies, there is a simple way to construct local information in the form of weak hierarchies from other local information (here given in the form of the map s) and to derive the desired global information directly from these locally defined weak hierarchies without further recourse to any other form of local information from which these locally defined weak hierarchies might have been deduced.

Similarly (cf. [BD92]), weakly compatible split systems $\mathcal{S}$ are defined to be those subsets $\mathcal{S}$ of $\mathcal{S}p(X)$ for which no three splits

$$S_1 = \{A_1, B_1\},\ S_2 = \{A_2, B_2\},\ S_3 \in \{A_3, B_3\} \in \mathcal{S}$$

with $A_1 \cap A_2 \cap A_3 \neq \emptyset$, $A_1 \cap B_2 \cap B_3 \neq \emptyset$, $B_1 \cap A_2 \cap B_3 \neq \emptyset$ and $B_1 \cap B_2 \cap A_3 \neq \emptyset$ exist or, equivalenty, for which

$$\mathcal{C}^x(\mathcal{S}) := \{A \subseteq X - \{x\} \mid \{A, X - A\} \in \mathcal{S}\}$$

is a weak hierarchy defined on $X - \{x\}$, for every $x \in X$. So, we have $\#\mathcal{S} \leq \binom{n}{2} + \binom{n}{1} + \binom{n}{0} = \binom{n+1}{2} + 1 = \binom{\#X}{2} + 1$ for every family $\mathcal{S}$ of weakly compatible splits (including possibly the trivial split $\{X, \emptyset\}$) defined on a set X of cardinality $n + 1$.

It follows easily from the results in [BD92] and [BD93] - or from the results regarding weak hierarchies just reported above and the relation between (affine) weak hierarchies and (projective) weakly compatible split systems - that also the theory of weakly compatible split systems fits excellently into the conceptual framework developed in §1. In particular, such split systems can also be viewed as resulting from the proposed "standard" procedure of extracting globally relevant from locally distributed information, provided that that locally distributed information conforms to some rather mild and easily specified requirements.

A certain generalization of weak hierarchies was discussed in [BD94], where a collection $\mathcal{C}$ of subsets of X was defined to be a *weak hierarchy of breadth at most* k if for all $C_1, C_2, \ldots, C_{k+1} \in \mathcal{C}$ one has

$$C_1 \cap C_2 \cap \cdots \cap C_{k+1} \in \{\bigcap_{i\neq j} C_i \mid j = 1, \ldots, k+1\}$$

or, equivalently, if there exist no clusters $C_1, \ldots, C_{k+1} \in \mathcal{C}$ and elements $x_1, \ldots, x_{k+1} \in X$ with $x_i \in C_j$ if and only if $i \neq j$. Clearly, $\mathcal{C} \subseteq \mathcal{P}(X)$ is a weak hierarchy of breadth at most k if and only if there exists at least one subset of cardinality k in any subset $Y \subseteq X$ of cardinality $k + 1$ which is not contained in $\mathcal{C}|_Y = \{C \cap Y \mid C \in \mathcal{C}\}$ - that is, if and only if $\mathcal{C}|_Y$ is a weak hierarchy of breadth at most k for every $Y \in \mathcal{P}_{\leq k+1}(X)$. Equivalently, $\mathcal{C} \subseteq \mathcal{P}(X)$ is a weak hierarchy of breadth at most k if and only if one has

$$\mathcal{P}(Y) = \{C_1 \cap C_2 \cap \cdots \cap C_\ell \cap Y \mid \ell \in \mathbb{N}_0,\ C_1, \ldots, C_\ell \in \mathcal{C}\}$$

for some $Y \subseteq X$ only if $\#Y \leq k$. As above, this implies easily

$$\#\mathcal{C} \leq \#\mathcal{P}_{\leq k}(X) = \binom{n}{k} + \binom{n}{k-1} + \binom{n}{k-2} + \cdots + \binom{n}{0}$$

for any such $\mathcal{C} \subseteq \mathcal{P}(X)$ (and $n := \#X$) as any $C \in \mathcal{C}$ necessarily contains a subset T of cardinality at most k with $C \subseteq C'$ for every $C' \in \mathcal{C}$ with $T \subseteq C'$.

Note also (cf. [BD94]) that a collection $\mathcal{C} \subseteq \mathcal{P}(X)$ of subsets of X is a weak hierarchy of breadth at most k if and only if, for any map $s : \mathcal{C} \to \mathbb{R}$ with $s(C) \leq s(C')$ for all $C, C' \in \mathcal{C}$ with $C \supseteq C'$, one has $\#A \leq k$ for any subset $A \subseteq X$ with

$$max(s(C) \mid A \subseteq C \in \mathcal{C}) < max(s(C') \mid A - \{a\} \subseteq C' \in \mathcal{C})$$

for all $a \in A$.

And it is also easy to see that given a weak hierarchy $i_Y \subseteq \mathcal{P}(Y)$ of breadth at most k for any subset Y of X of cardinality at most $k+1$ which is closed with respect to intersection and contains Y as well as the empty set $\emptyset$, we have $res_{X \to Y}(\bar{i}_X) = i_Y$ for all $Y \in \mathcal{P}_{\leq k+1}(X)$ if and only if we have $res_{Z \to Y}(\bar{i}_Z) = i_Y$ for all $Z \subseteq X$ of cardinality at most $2k+1$ and all $Y \in \mathcal{Y} \cap \mathcal{P}(Z)$ (with $\bar{i}_Z$ defined relative to the i_Y for all Z in X just as above, of course).

And finally, given a weak hierarchy $\mathcal{C} \subseteq \mathcal{P}(X)$ of breadth at most k, we always have $\mathcal{C} \subseteq \bar{i}_X \subseteq \hat{\mathcal{C}}$ for the weak hierarchy $\bar{i}_X$ of breadth at most k associated with the family $i_Y := \mathcal{C}|_Y$ $(Y \in \mathcal{P}_{\leq k+1}(X))$ and the smallest intersection-closed subset $\hat{\mathcal{C}}$ of $\mathcal{P}(X)$ containing $\{X\} \cup \mathcal{C}$ which in this case coincides with

$$\{X\} \cup \{C_1 \cap \cdots \cap C_k | C_1, \cdots, C_k \in \mathcal{C}\}.$$

Moreover, $\bar{i}_X = \hat{\mathcal{C}}$ holds if and only if i_Y contains Y and is closed with respect to pairwise intersection for each $Y \in \mathcal{P}_{\leq k+1}(X))$ which in turn is surely the case if X and all subsets of cardinality at most $k-1$ belong to $\mathcal{C}$.

To prove these statements, it is enough to show that $\mathcal{C} = \bar{i}_X$ holds if $\mathcal{C}$ coincides with $\hat{\mathcal{C}}$. Otherwise, there would exist some subset $A \subseteq X$ with $A \in \bar{i}_X \backslash \mathcal{C}$ and, hence, there would also exist some smallest subset Z of X with $A \cap Z \notin \mathcal{C}|_Z$. It follows that for any $z \in Z$, there would exist some $C_z \in \mathcal{C}$ with $A \cap (Z - \{z\}) = C_z \cap (Z - \{z\})$ as well as $A \cap Z \neq C_z \cap Z$ and, hence, with

$$C_z \cap Z = \begin{cases} (A \cap Z) - \{z\} \text{ if } z \in A \cap Z, \\ (A \cap Z) \cup \{z\} \text{ else.} \end{cases}$$

While the first assertion implies $\#(A \cap Z) \leq k$ in view of the assumption that $\mathcal{C}$ is a weak hierarchy of breadth at most k, the second assertion implies $(Z \backslash A) \leq 1$ because $z_1, z_2 \in Z \backslash A$ and $z_1 \neq z_2$ would imply $C_{z_1} \cap C_{z_2} \cap Z = A \cap Z$ in contradiction to $C_{z_1} \cap C_{z_2} \in \mathcal{C}$ and our choice of A and Z. So, we would have $\#Z \leq k+1$, now in contradiction to the fact that this implies $A \cap Z \in i_Z = \mathcal{C}|_Z$ for all $A \in \bar{i}_X$ in view of the definition of $\bar{i}_X$.

3. Some Upper Bounds Regarding More General Cluster Systems

Finally, we want to justify the rather general framework for clustering theory presented in the first section by establishing the following rather general result which implies most of the inequalities mentioned above:

Theorem. *Given a collection $\mathcal{C} \subseteq \mathcal{P}(X)$ of subsets of a finite set X and a simplicial complex $\mathcal{X} \subseteq \mathcal{P}(X)$ of subsets of X (that is, a collection of subsets of X*

which is closed relative to taking subsets, that is, $A \subseteq B$ and $B \in \mathcal{X}$ implies $A \in \mathcal{X}$), such that every subset A of X with $\mathcal{P}(A) = \{A \cap C \mid C \in \mathcal{C}\}$ is contained in $\mathcal{X}$, then we have

$$\#\mathcal{C} \leq \#\mathcal{X}.$$

Proof: For each $C \subseteq X$, consider the map

$$\chi_C : \mathcal{X} \to \mathbb{R} : A \mapsto (-1)^{\#A \cap C}.$$

All we need to show is that the maps χ_C with C in $\mathcal{C}$ are linearly independent which we do by induction on $n := \#X$. Clearly, our assumption holds in case $n = 0$. So assume that we have coefficients $r_C \in \mathbb{R}$ $(C \subseteq X)$ with $r_C = 0$ for $C \notin \mathcal{C}$ and $\sum_{C \subseteq X} r_C \chi_C(A) = 0$ for all $A \in \mathcal{X}$. Choose some arbitrary $x \in X$, consider $\mathcal{C}_x := \{C \backslash \{x\} \mid C \in \mathcal{C}\} \subseteq X - \{x\}$ and note that $\{Y \cap C \mid C \in \mathcal{C}_x\} = \{Y \cap C \mid C \in \mathcal{C}\}$ for all $Y \subseteq X - \{x\}$, so we have $Y \in \mathcal{X}_x := \{A \subseteq X - \{x\} \mid A \in \mathcal{X}\}$ for all $Y \subseteq X - \{x\}$ with $\{Y \cap C \mid C \in \mathcal{C}_x\} = \mathcal{P}(Y)$. As $\sum_{C \subseteq X} r_C \chi_C(A) = 0$ for all $A \in \mathcal{X}$ implies $\sum_{C \subseteq X - \{x\}} (r_C + r_{C \cup \{x\}}) \chi_C(A) = 0$ for all $A \in \mathcal{X}_x$ and as $r_C + r_{C \cup \{x\}} = 0$ whenever $C \notin \mathcal{C}_x$, our induction hypothesis implies

$$r_C = -r_{C \cup \{x\}}$$

for all $x \in X$ and $C \subseteq X - \{x\}$. Hence, if $r_{C_0} \neq 0$ for one subset $C_0 \subseteq X$, a simple induction argument based on the cardinality of $(C \backslash C_0) \cup (C_0 \backslash C)$ would imply $r_C \neq 0$ for all $C \subseteq X$ and therefore $\mathcal{C} = \mathcal{P}(X)$ which in turn would imply $\mathcal{X} = \mathcal{P}(X)$ and, therefore, $\sum_{C \in \mathcal{C}} r_C \chi_C(X) = 0$ with

$$r_C = (-1)^{\#C} r_\emptyset \neq 0$$

for all $C \subseteq X$ in contradiction to

$$\sum_{C \in \mathcal{C}} r_C \chi_C(X) = \sum_{C \in \mathcal{C}} r_\emptyset \cdot (-1)^{\#C} \cdot (-1)^{\#(C \cap X)} = 2^n \cdot r_\emptyset \neq 0.$$

□

This result - and its "projective" analogue - clearly presents a far-reaching generalization of (some of) the results recalled in the previous section. It shows that we should be able to compute in reasonable time the cluster systems $\bar{i}_X$ related to systems of "local" information $i_Y (Y \in \mathcal{Y})$ provided the simplicial complex $\mathcal{X}$ consisting of all $A \subseteq X$ with $\bar{i}_A = \mathcal{P}(A)$ is of a reasonable size (with $\bar{i}_A := \{A' \subseteq A \mid A' \cap Y \in i_Y$ for all $Y \in \mathcal{Y}\}$, as above).

We will not work out the consequences of this result here. Rather, we refer to forthcoming papers with H.J. Bandelt, V. Chepoi and J. Koolen where in particular those *extremal* collections $\mathcal{C}$ of subsets of X will be studied for which $\#\mathcal{C} = \#\mathcal{X}$ holds for the simplicial complex $\mathcal{X} = X(\mathcal{C})$ which consists of all subsets $Y \subseteq X$ with $\mathcal{P}(Y) = \{C \cap Y \mid C \in \mathcal{C}\}$, - just noting that, in case a collection $\mathcal{C}$ of subsets of X is closed with respect to intersection and contains X, it satisfies the condition $\#\mathcal{C} = \#\mathcal{X}(\mathcal{C})$ if and only if it is a convex geometry as defined in [ED85] (or - equivalently - an anti matroid).

Rather, we close with the following observation: As much stronger bounds hold for hierarchies than for weak hierarchies, one might be tempted to believe that given a subset $i_Y \subseteq \mathcal{P}(Y)$ for, say, each $Y \in \mathcal{P}_{\leq 3}(X)$ with $\#i_Y \leq \#\mathcal{P}(Y) - 2$ for each $Y \in \mathcal{P}_3(X)$, the cardinality of the resulting set $\bar{i}_X := \{A \subseteq X \mid A \cap Y \in i_Y$ for all $Y \in \mathcal{P}_{\leq 3}(X)\}$ should also be considerably smaller than that of $\mathcal{P}_{\leq 2}(X)$, the upper bound we get immediately from the above theorem. Yet the example $X := \{1, 2, \ldots, 2n\}$ and

$$i_Y := \begin{cases} \mathcal{P}(Y) \text{ if } \#Y \leq 2 \\ \{\emptyset\} \cup \{\{i,j\} \subseteq Y \mid i,j \leq n \text{ or } n+1 \leq i,j\} \text{ if } \#Y = 3 \end{cases}$$

which leads to

$$\bar{i}_X = \{\emptyset\} \cup \{\{i,j\} \subseteq X \mid i,j \leq n \text{ or } n+1 \leq i,j\}$$

shows that such an expectation would not be justified.

Instead, given a simplicial complex $\mathcal{X}$ of subsets of X, one might define a system $\mathcal{C}$ of subsets of X to be an $\mathcal{X}$-hierarchy if $C_1 \cap C_2 \in \mathcal{X}$ holds for all $C_1, C_2 \in \mathcal{C}$ for which neither $C_1 \subseteq C_2$ nor $C_2 \subseteq C_1$ holds. Clearly, if $\mathcal{C}$ is an $\mathcal{X}$-hierarchy, then so is any subset of $\mathcal{C}$ as well as the set $\mathcal{C} \cup \{X\} \cup \mathcal{X}^*$ with $\mathcal{X}^*$ defined by

$$\mathcal{X}^* := \{A \subseteq X \mid \text{every proper subset of } A \text{ belongs to } \mathcal{X}\};$$

so in particular, the smallest subset $\hat{\mathcal{C}}$ of $\mathcal{P}(X)$ containing a given $\mathcal{X}$-hierarchy $\mathcal{C}$ and being closed with respect to intersection is always an $\mathcal{X}$-hierarchy. It is also clear that hierarchies are just $\{\emptyset\}$-hierarchies. And it follows easily from the above result regarding hierarchies that we have

$$\#(\mathcal{C} \backslash \mathcal{X}^*) < \#\mathcal{A}$$

for the set

$$\mathcal{A} := \{A \subseteq X \mid A \notin \mathcal{X} \text{ and } A - \{a\} \in X \text{ for all but (at most) one } a \in A\},$$

because associating to any $C \in \mathcal{C} \backslash \mathcal{X}^*$ the subset $\mathcal{A}(C) := \mathcal{P}(C) \cap \mathcal{A}$ of $\mathcal{A}$ produces an $\mathcal{A}$-hierarchy $\mathcal{C}' := \{\mathcal{A}(C) \mid C \in \mathcal{C}\}$ whose cardinality $\#\mathcal{C}'$ coincides with that of $\mathcal{C} \backslash \mathcal{X}^*$ in view of $C = \bigcup_{A \in \mathcal{A}(C)} A$ for all $C \in \mathcal{C} \backslash \mathcal{X}^*$, which consists of subsets of $\mathcal{A}$ each containing at least two distinct elements from $\mathcal{A}$.

Of course, the hierarchy $\mathcal{C}'$ is in general far from being binary and, hence, its cardinality will be *considerably* smaller than that of $\mathcal{A}$. Yet, the example $X := \mathbb{P}_d(\mathbb{F}_2) := \mathbb{F}_2^{d+1} - \{0\}$, the d-dimensional projective space over $\mathbb{F}_2$, and $\mathcal{C} := \{U - \{0\} \mid U$ a subspace of $\mathbb{F}_2^{d+1}$ of dimension $2\}$, the set of lines in $\mathbb{P}_d(\mathbb{F}_2)$, provides an example of an $\mathcal{X}$-hierarchy for $\mathcal{X} := \mathbb{P}_{\leq 1}(X)$ of cardinality $(2^{d+1} - 1)(2^d - 1)/3$ defined on a set of cardinality $2^{d+1} - 1$ which shows that at least the order of magnitude of $\mathcal{C} \backslash \mathcal{X}^*$ is described correctly by the above bound. Still, it is probably quite an interesting problem to study the extremal $\mathcal{X}$-hierarchies $\mathcal{C}$ (that is, those $\mathcal{X}$-hierarchies $\mathcal{C}$ which have the largest possible cardinality among all $\mathcal{X}$-hierarchies) in some detail, - at least in the case of "k-hierarchies", that is the $\mathcal{P}_{\leq k}(X)$-hierarchies, for which the above - and surely improvable - bound gives

$$\#(\mathcal{C} \backslash \mathcal{P}_{\leq k+1}(X)) < \binom{n}{n+1}.$$

Still more generally, for any two simplicial complexes $\mathcal{X}_1$ and $\mathcal{X}_2$ consisting of subsets of X, we may define a cluster system $\mathcal{C}$ contained in $\mathcal{P}(X)$ to be an $(\mathcal{X}_1, \mathcal{X}_2)$-hierarchy if and only if for all $k \in \mathbb{N}$ and $C_1, \ldots, C_k \in \mathcal{C}$ we have (non-exclusively) either (i) $C_1 \cap \cdots \cap C_k \in \mathcal{X}_1$ or (ii) $C_1 \cap \cdots \cap C_k = C_1 \cap \cdots \cap C_{i-1} \cap C_{i+1} \cap \cdots \cap C_k$ for some $i \in \{1, \ldots, k\}$ or (iii) $\{a_1, \ldots, a_k\} \in \mathcal{X}_2$ for all $a_1, \ldots, a_k \in X$ with $a_i \in C_j$ if and only if $i \neq j$, for all $i, j = 1, \ldots, k$. In particular, we may define $\mathcal{C}$ to be a (k, ℓ)-hierarchy for any two integers $k, \ell \geq -1$ if and only $\mathcal{C}$ is an $(\mathcal{P}_{\leq k}(X),\ \mathcal{P}_{\leq \ell}(X))$-hierarchy, that is if and only if for all $C_1, \ldots, C_{\ell+1} \in \mathcal{C}$ we have $\#(C_1 \cap C_2 \cap \cdots C_{\ell+1}) \leq k$ or $C_1 \cap C_2 \cap \cdots \cap C_{\ell+1} = C_1 \cap \cdots \cap C_{i-1} \cap C_{i+1} \cap \cdots \cap C_{\ell+1}$ for some $i \in \{1, \ldots, \ell+1\}$. Using this terminology, it is easy to see that a hierarchy $\mathcal{C}$ as defined in **2.1** is just a $(0,1)$-hierarchy, while an $\mathcal{X}$-hierarchy is an $(\mathcal{X}, \mathcal{P}_{\leq 1}(X))$-hierarchy, a weak hierarchy is a $(-1,2)$-hierarchy, and a weak hierarchy of breadth at most ℓ is a $(-1,\ell)$-hierarchy. Note also that the almost obvious fact that every hierarchy is a weak hierarchy (which actually - ten years ago - presented the motivation for naming them that way) now generalises to the simple lemma that every (k,ℓ)-hierarchy is a $(k-1, \ell+1)$-hierarchy.

It is left to the interested reader to establish that a cluster system $\mathcal{C} \subseteq \mathcal{P}(X)$ is a (k,ℓ)-hierarchy if and only if $\mathcal{C}|_Y$ contains at most ℓ subsets of cardinality $k+\ell+1$ for any subset Y of X of cardinality $k+\ell+2$, to find useful upper bounds regarding the number of clusters in an $(\mathcal{X}_1, \mathcal{X}_2)$-hierarchy $\mathcal{C}$ by using the theorems proved above or to search for even better bounds as well as to translate all that from the affine to the projective case. All that I wanted to establish (and hope to have established by now) is that viewing clustering techniques from the point of view proposed in the first section of this note, does not only allows one to put a large body of known results into a uniform conceptual framework but also leads to a considerable number of new and interesting results and data-analysis tools.

Acknowledgement

This paper was written while the author was hosted by the Biomathematics Research Centre at the Mathematics Department, University of Canterbury, New Zealand, which is gratefully acknowledged. Many thanks go in particular to Ann Tindall for her careful typesetting and to Mike Steel for many useful comments. Many thanks go also to Boris Mirkin for his steadfast encouragements to actually work out and put down the ideas presented here without which encouragments this note would probably never have been written.

References

[BD86] H.-J. Bandelt and A. Dress, *Reconstructing the shape of a tree from observed dissimilarity data*, Advances in Applied Mathematics, Vol. 7 (1986), 309-343.

[BD89] H.-J. Bandelt and A. Dress, *Weak hierarchies associated with similarity measures – an additive clustering technique,* Bull. Math. Biology, Vol. 51 (1989), 133-166.

[BD92] H.-J. Bandelt and A. Dress, *A canonical decomposition theory for metrics on a finite set*, Advances in Mathematics, 92(1992), 47-105.

[BD94] H.-J. Bandelt and A. Dress, *An order theoretic framework for overlapping clustering*, *Discrete Mathematics*, **136**, (1994), 21-37.

[BD93] H.-J. Bandelt and A. Dress, *A relational approach to split decomposition*; *Information and Classification*, O. Opitz, B. Lausen, R. Klar editors, Berlin; Springer-Verlag, (1993), 123-131.

[B71] P. Buneman, *The recovery of trees from measures of dissimilarity*, In: Mathematics in the Archaeological and Historical Sciences, F.Hodson, D. Kendall, and P. Tautu (Ed.s) Edinburgh University Press, Edinburgh, 1971, 387-395.

[DM85] W.H.E. Day and F.R. McMorris, (1985), A formalization of consensus index methods, *Bull. Math. Biol.* **47**, 215-229.

[ED85] P.H. Edelman and R. Jamison, *The Theory of Convex Geometries*, *Geom. Dedicata* **21** (1985), 247-274.

[G87] A.D. Gordon, (1987) *A review of hierarchical classification*, J. Royal Statistical Society Ser. A 150, 119-137.

[M97] B. Mirkin, this volume.

UNI BIELEFELD/FSPM POSTFACH 10 01 31, 33501, BIELEFELD, GERMANY

E-mail address: dress@Mathematik.Uni-Bielefeld.DE

This page intentionally left blank

https://doi.org/10.1090/dimacs/037/20

DIMACS Series in Discrete Mathematics
and Theoretical Computer Science
Volume **37**, 1997

On Hierarchies and Hierarchical Classes Models

Iven Van Mechelen, Seymour Rosenberg, and Paul De Boeck

ABSTRACT. A hierarchical classification typically refers to a sequence of nested partitions of an object set. However, this type of mathematical formalization of the concept of hierarchy is neither the only extant one in the classification domain nor the most general one. Indeed, alternative formalizations of the concept of hierarchy were introduced early on in the literature of numerical taxonomy. In this paper we will give a definition of hierarchy that generalizes the notion of a sequence of nested partitions. A family of hierarchical classification models based on this definition (viz. the family of hierarchical classes models introduced by De Boeck and Rosenberg, 1988, and Van Mechelen, De Boeck and Rosenberg, 1995) will be briefly described.

1. Definitions of Hierarchical Classification

Rather than presenting a review of how the classification domain has dealt with the definition of hierarchy during the last few decades, we will limit ourselves to outlining three main issues on which different definitions diverge. In order to present these three issues, it is sufficient to refer to the definitions of hierarchy found in the seminal works of Sneath and Sokal (1973) and Jardine and Sibson (1971).

We assume that some set of objects (elements, operational taxonomic units or OTU's, ...) is given. All definitions of hierarchy in the classification domain imply a sequence of clusterings of the object set under consideration. At this point a clustering may be simply thought of as any family of sets of objects. The sequence of clusterings may be considered to be indexed by some ordinal or numerical level. If each clustering is exhaustive in that it spans the entire object set (which we do not assume at this moment), Jardine and Sibson (1971) call the sequence a *stratified* clustering.

The three issues referred to above are: (1) Are the clusterings at each level of the sequence allowed to be overlapping or not? (2) Are clusterings at different levels necessarily nested in one another or not? (3) What constitutes the key principle of the hierarchical organization? Each of these issues is taken up, in turn, in the next three sections.

1.1. Nonoverlapping vs. overlapping clusterings. As already noted, a hierarchical classification typically refers to a sequence of clusterings in which there

1991 *Mathematics Subject Classification.* 62H30.

The first two authors were supported in part by NATO Grant CRG.921321.

The authors thank Boris Mirkin and Iwin Leenen for their helpful comments on a previous version of this paper.

is *no overlap* among the clusters at each of the different levels of the hierarchy. When each clustering also spans the entire object set, the hierarchical classification is a sequence of partitions of the object set.

When introducing their definition of a hierarchic clustering, however, Sneath and Sokal (1973) state that

> "at any level of clustering there is no limitation on the degree of overlap of taxa, that is, one OTU may be simultaneously a member of two or more taxa" (p. 205).

To justify this definition, they further point out that partitions often distort the phenetic relationships (i.e., the relationships based on similarity in attributes or characters) among OTU's.

For their part, Jardine and Sibson (1971) limit the term hierarchic to nonoverlapping stratified clusterings. In their mathematical formalization of hierarchical clustering, they further deal with the partitions at each level of the hierarchy in terms of the corresponding equivalence relations (the clusters being the respective equivalence classes). Jardine and Sibson have generalized the latter formalization to include also models for overlapping stratified clustering. However, they do not want to use the term hierarchic to designate the latter type of models. Jardine and Sibson's formalization of overlapping stratified clusterings is based on a generalization of the equivalence relations mentioned above, which are reflexive, symmetric and transitive, to relations that are reflexive and symmetric only. The clusters then are the maximal object sets all pairs of which belong to the relation. (In graph theoretic terms these maximal sets correspond to maximal cliques.) Note that, unlike Sneath and Sokal's position to put no limitations on the type of overlap, the formalization in terms of reflexive and symmetric relations does imply such limitations. For example, it precludes a clustering two clusters of which are in a subset-superset relation; also, it precludes clusterings that include all possible pairs of objects out of a subset of three or more objects (Jardine and Sibson, 1971).

1.2. Nested vs. non-nested clusterings. A sequence of clusterings is called *nested* if for any two clusterings it holds that the lower level clustering is a refinement of the higher level clustering, that is, if any cluster of the lower level clustering is a subset of (at least) one cluster of the higher level clustering.

In the prototype of hierarchical classification models, clusterings are assumed to be nested. For Jardine and Sibson (1971), nestedness is a necessary feature of both nonoverlapping and overlapping stratified clusterings; in their formalization, the nestedness condition follows from the requirement that the (equivalence resp. reflexive and symmetric) relation corresponding to a lower level clustering should always be a subset of the (equivalence resp. reflexive and symmetric) relation corresponding to any higher order clustering.

In contrast, Sneath and Sokal (1973) include non-nested clusterings in their definition of hierarchy:

> "OTU's that are members of a common taxon at a lower level may again be members of different taxa at a higher level" (pp. 205-206).

Sneath and Sokal note that this is the common dictionary meaning of hierarchy.

1.3. Key principle of hierarchic organization. As indicated above, any hierarchical classification implies a sequence of clusterings indexed by some numerical or ordinal level function. Sneath and Sokal (1973) list two properties linked

to the hierarchical level, the second of which suggests a key principle behind any hierarchical organization. The first property reads that the number of clusters in the successive clusterings of the sequence should be nonincreasing. The second property is formulated by Sneath and Sokal in a negative way:

> "Classifications that are *nonhierarchic* are those that do not exhibit ranks in which subsidiary taxa become members of larger more inclusive taxa. Set-theoretically, the OTU's do not exhibit partial order." (p. 206, italics theirs).

Hence, in this view a central role is given to partial orders.

Three points deserve clarification: First, it seems plausible that the partial order Sneath and Sokal are referring to is the partial order defined on the family of all clusters of a hierarchic clustering based on the relation of set inclusion; this, however, is a partial order defined on the clusters rather than on the OTU's. Second, in principle Sneath and Sokal's definition of hierarchy allows for any type of overlap at all levels of a hierarchic clustering; this gives room for subset-superset relations between clusters at a same hierarchic level, which would contradict partial order based on set inclusion as the organizational principle of the hierarchy. Third, in general, the number of clusters at the same level may increase when going higher up in a partial order; the latter could contradict the first property listed by Sneath and Sokal.

For their part, Jardine and Sibson (1971), as already mentioned above, require that the (equivalence resp. reflexive and symmetric) relation corresponding to a lower level clustering should always be a subset of the (equivalence resp. reflexive and symmetric) relation corresponding to any higher order clustering. For stratified nonverlapping clusterings, this implies that the number of clusters in the hierarchy should be nonincreasing, indeed; for stratified overlapping clusterings, however, this is not necessarily the case. With respect to the partial order basis of the hierarchic organization, this holds in the case of both nonoverlapping and overlapping stratified clusterings.

1.4. Discussion. From the above it becomes clear that various formalizations of the concept of hierarchy are possible in the classification domain. Perhaps the most restrictive definition is that of a sequence of nonoverlapping, nested clusterings. If in addition each clustering spans the entire object set and if the highest clustering consists of a single cluster (i.e. the entire object set) the clustering meets Jardine and Sibson's definition of a hierarchic stratified clustering. In case the clustering sequence of a hierarchic stratified clustering is indexed by an ordinal level, it is denoted by Jardine and Sibson as a *taxonomic* or *Linnaean* hierarchy; if it is indexed by some numerical level, it is denoted by them as a *dendrogram.* Somewhat less restrictive concepts of hierarchy have been suggested more recently as those of weak hierarchy (Bandelt and Dress, 1989) or pyramid (Diday, 1986) (for a review, see Mirkin, 1996).

On the other hand, the most general definition may be that of a possibly nonnested sequence of possibly overlapping clusterings, with the partial order relation of the clusters defined on the basis of set inclusion as organizational basis of the hierarchy. In order to yield a consistent definition of hierarchical relations, the clusterings at the different hierarchic levels should further be restricted such that no clustering includes a cluster that is a subset of another cluster at the same level or at a lower level of the hierarchy. In the most general case there is also no need

to require that the number of clusters should be nonincreasing when going up in the hierarchy.

In the next part of this paper we will describe a family of classification models based on this general definition of hierarchy, namely the family of hierarchical classes models.

2. The Family of Hierarchical Classes Models

In this section we first briefly introduce the basic hierarchical classes model, an extension, and the associated data analysis. Second we discuss how the family of hierarchical classes models relates to the three issues outlined in the previous section.

2.1. Models and data analysis.

2.1.1. *Basic model.* Hierarchical classes models are structural models for two-way two-mode arrays M with binary (0,1) entries. The elements of the first mode are called objects; depending on the substantive application they may refer to biological species, persons etc. The elements of the second mode are called attributes. Table 1 contains a hypothetical array with 7 objects and 8 attributes.

TABLE 1. A Hypothetical Two-Way Two-Mode Matrix

	Attributes							
Objects	a	b	c	d	e	f	g	h
1	1	0	0	1	1	1	0	1
2	0	1	1	0	0	1	1	1
3	0	1	1	0	0	1	1	1
4	0	0	0	1	1	0	0	1
5	0	0	0	0	0	1	0	1
6	0	0	0	0	0	0	1	0
7	0	0	0	0	0	0	1	0

Various methods could of course be applied to derive proximities from a binary two-way two-mode matrix, and those proximities could then be subjected to standard hierarchical clustering methods. In the approach presented here, however, the model will represent directly the two-way two-mode matrix rather than derived proximities.

Given a binary matrix like that of Table 1, natural clusters of objects may be defined as the sets of objects to which each of the attributes applies. Note that different attributes may yield the same object cluster. Each cluster can further be labeled by its defining attribute(s).

A partial order hierarchy can be defined on the obtained clusters based on set inclusion. This partial order induces a quasi-order (i.e., a reflexive and transitive relation) on the labeling attributes. The hierarchy of the object clusters can be graphically represented by a Hasse diagram. A Hasse diagram of the hierarchy of object clusters derived from Table 1 is presented in Figure 1(a).

Similarly, one could define a natural hierarchy of labeled attribute clusters, defined as the sets of attributes that apply to each of the objects. A Hasse diagram of the hierarchy of attribute classes derived from Table 1 is presented in Figure

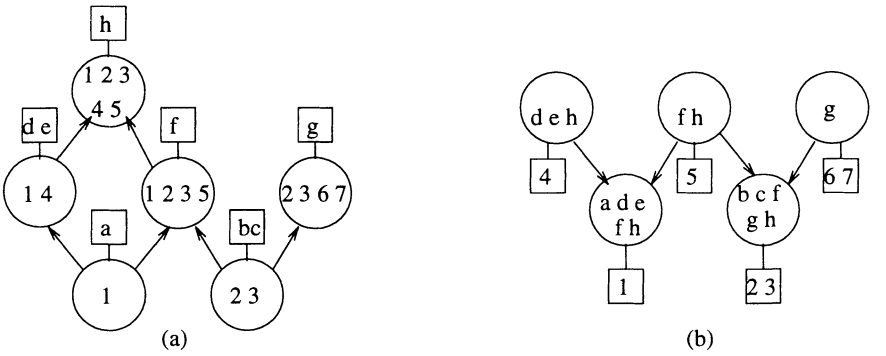


FIGURE 1. Hasse diagrams of hierarchies of object clusters (a) and attribute clusters (b) derived from Table 1.

1(b). For reasons that will become clear soon, the Hasse diagram is drawn upside down.

A hierarchical classes model of a binary $m \times n$ matrix M implies a Boolean decomposition of M into a binary $m \times k$ matrix S and a binary $n \times k$ matrix P, where k denotes the rank of the model (De Boeck and Rosenberg, 1988; Van Mechelen, De Boeck and Rosenberg, 1995). The columns of S and P are called object and attribute bundles, respectively. Several decomposition rules may be distinguished. For example, Van Mechelen, De Boeck, and Rosenberg (1995) describe the following conjunctive decomposition rule:

$$M = [S^c \otimes P']^c \tag{1}$$

where $'$ denotes transpose, ^{c} complement, and $\otimes$ the Boolean matrix product. The important point here is that the bundle matrices S and P of any hierarchical classes model are required to represent (directly or inversely) the hierarchies of object and attribute clusters as defined above. In particular, for the conjunctive hierarchical classes model with decomposition rule (1), the natural object hierarchy that may be derived from S should be identical to the object hierarchy that may be derived from M; furthermore, the natural attribute hierarchy that may be derived from P should be the inverse of the attribute hierarchy that may be derived from M.

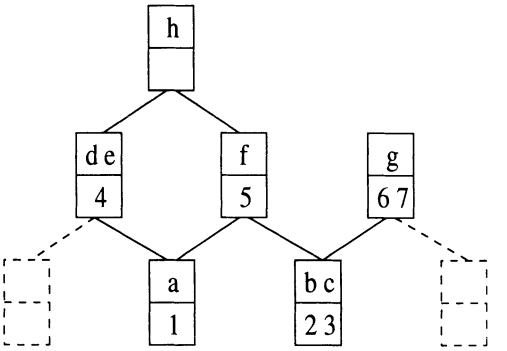


FIGURE 2. (Conjunctive) hierarchical classes model for data of Table 1.

A hierarchical classes model can be given a graphic representation that gives a comprehensive account of the model. Figure 2 contains a graphic representation of the conjunctive model with decomposition rule (1) for the matrix M of Table 1. The nonempty boxes in the graphic representation of Figure 2 correspond exactly to the boxes with labels of Figures 1 (a) and 1 (b); the dashed boxes at the bottom of the representation refer to the object and attribute bundles. The cluster of objects labeled by a given attribute can further be easily derived from the figure as the set of all objects below the attribute in the representation; similarly, the cluster of attributes labeled by a given object corresponds to the set of all attributes above that object in the graphic representation.

2.1.2. *Extension.* The object clusters of the hierarchical classes models are the clusters that are naturally defined by each of the attributes in the two-way two-mode matrix M; an analogous definition was used for the attribute clusters. This definition of clusters could be extended to clusters that are defined by all possible conjunctions of attributes (resp. objects). The set of all object clusters that arises in this way can again be partially ordered on the basis of set inclusion. The resulting partially ordered set of object clusters can be shown to be a lattice, which, moreover, is anti-isomorphic to the lattice of attribute clusters. This lattice is known as the Galois lattice of the binary relation defined by the matrix M (Barbut and Monjardet, 1970; Birkhoff, 1940).

The Galois lattice meets our general definition of a hierarchical classification. From the point of view of the clusters, hierarchical classes models may be considered subsets of corresponding Galois lattices.

2.1.3. *Data analysis.* Most binary data matrices D can be perfectly represented only by very complex hierarchical classes models. One therefore generally will look for binary matrices M that approximately equal D and that can be represented by a parsimonious hierarchical classes model.

Data analysis on the basis of a hierarchical classes model starts by specifying a bound on the complexity of the desired model. This specification is done in terms of the matrix decomposition implied by the model. An algorithm exists to look for a matrix M that approximates D as closely as possible and that can be represented by a hierarchical classes model of the desired complexity; in this, close approximation is to be understood in the least squares sense.

Unlike hierarchical classes models Galois lattices do not imply a matrix decomposition. The Galois lattice approach, however, may be extended such as to include the matrix decomposition implied by the hierarchical classes approach (Van Mechelen, 1993). This extension makes it possible to construct parsimonious approximate Galois lattices, which makes lattice analysis amenable to real data.

2.2. Type of hierarchy. We will now successively examine the position of the hierarchical classes family vis-a-vis the three issues discussed in the first part of this paper. With respect to the nonoverlapping-overlapping issue, at any level of the hierarchy with more than a single cluster, the clustering may be an overlapping one. For example, at the second level of the hierarchy of object clusters represented in Figure 1 (a), there is overlap between the clusters 1,2,3,5 and 2,3,6,7. Unlike the position of Sneath and Sokal (1973), there are some limitations as regards the type of overlap that is allowed in a hierarchical classes model. In particular, clusters at the same hierarchical level are not allowed to be in a subset- superset relation. The limitations, however, are less strict than those advanced by Jardine and Sibson

(1971), in that the possible overlapping clusterings are not restricted to maximal cliques of reflexive and symmetric relations. For example, the hierarchical classes approach leaves room for clusterings that include all possible pairs of objects out of a subset of three or more objects.

With respect to the *nestedness* issue, in general the hierarchies in hierarchical classes models are nonnested nesting being a special case. For example, the cluster 2,3,6,7 at the second level of the hierarchy of Figure 1 (a) is not a subset of the single cluster 1,2,3,4,5 at the third level of the hierarchy.

With respect to the *key principle of hierarchical organization*, the hierarchical classes approach obviously is fully based on the partial order principle. Furthermore, unlike Sneath and Sokal's definition, the number of clusters at a same hierarchical level may increase when going up in the hierarchy. For example, in the hierarchy of Figure 1 (a), there are two clusters on the first hierarchical level and three clusters on the second.

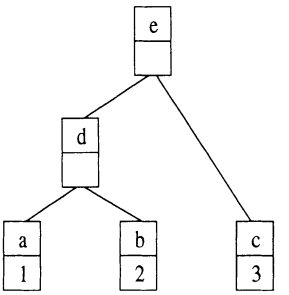


FIGURE 3. Hierarchical classes model of a hypothetical Linnaean hierarchy.

In general the definition of hierarchy used in the hierarchical classes approach is a generalization of the prototypical definition in the classification domain. It includes a nested hierarchy of nonoverlapping clusters as a special case. This is illustrated by Figure 3 that represents a hierarchical classes model of a simple Linnaean hierarchy.

3. Concluding Remarks

In this paper we reviewed the fact that the concept of hierarchical classification has been given different mathematical formalizations early on in the development of numerical taxonomy. The notion of a partial order defined in terms of set inclusion appears to be a recurrent key principle of hierarchic organization in several approaches. The example of the family of hierarchical classes models, which we briefly described in the second part of this paper, illustrates one possible way to elaborate this key principle within a classification model in a consistent and general way.

The hierarchical classes example also illustrates that it is possible to represent two-way two-mode data by a clustering model *without* first converting these data into two-way one-mode proximities. As such, it is an instance of the broader family of direct clustering methods. Another distinctive feature of the hierarchical classes approach is the fact that it implies both a clustering of objects and a clustering of attributes. As such the approach fully respects the two-sided nature of the data,

the hierarchical classes models being a member of the broader class of two-sided clustering models.

References

[1] Barbut, M., and Monjardet, B. (1970). *Ordre et Classification: Algebre et Combinatoire (2 Vols.)*. Paris: Hachette.

[2] H.-J. Bandelt and A.W.M. Dress (1989) Weak hierarchies associated with similarity measures – an additive clustering technique, *Bulletin of Mathematical Biology, 51*, 133-166.

[3] Birkhoff, G. (1940). *Lattice Theory*. Providence: American Mathematical Society.

[4] E. Diday (1986) Orders and overlapping clusters by pyramids. In: J. de Leeuw, W. Heiser, J. Meulman, and F. Critchley (Eds.) *Multidimensional Data Analysis*, Leiden: DSWO Press, 201-234.

[5] De Boeck, P., and Rosenberg, S. (1988). Hierarchical classes: Model and data analysis. *Psychometrika, 53*, 361-381.

[6] Jardine, N., and Sibson, R. (1971). *Mathematical Taxonomy*. London: Wiley.

[7] Mirkin, B. (1996) *Mathematical Classification and Clustering*. Boston-Dordrecht: Kluwer Acedemic Press.

[8] Sneath, P. H. A., and Sokal, R. R. (1973). *Numerical Taxonomy*. San Francisco: Freeman.

[9] Van Mechelen, I. (1993). Approximate Galois lattices of formal concepts. In O. Opitz, B. Lausen, and R. Klar (Eds.) *Information and classification: Concepts, methods and applications* (pp. 108-112). Berlin: Springer.

[10] Van Mechelen, I., De Boeck, P., and Rosenberg, S. (1995). The conjunctive model of hierarchical classes. *Psychometrika, 60,* 505-521.

Department of Psychology,, Katholieke Universiteit Leuven, Tiensestraat 102,, B-3000 Leuven, Belgium

E-mail address: Iven.VanMechelen@psy.kuleuven.ac.be

Rutgers University, NJ, USA

Katholieke Universiteit Leuven, Tiensestraat 102,, B-3000 Leuven, Belgium

https://doi.org/10.1090/dimacs/037/21

DIMACS Series in Discrete Mathematics
and Theoretical Computer Science
Volume **37**, 1997

The Construction of Globally Optimal Ordered Partitions

Lawrence Hubert, Phipps Arabie, and Jacqueline Meulman

ABSTRACT. The classification task discussed is one of constructing for an object set S an optimal partition into a given number of ordered classes based on symmetric or skew-symmetric proximity information among the objects. Several measures of merit for how well a given ordered partition reflects the proximity data are defined, and a dynamic programming strategy is suggested for their maximization. A numerical illustration is provided using a published data set for how the process can be carried out. A few generalizations are mentioned, including the construction of a median for a collection of proximity matrices that represents an ordered partition and how the use of ordered partitions might be one mechanism for generalizing the notion of an ultrametric.

1. Introduction

The classification task considered in the present paper is the construction of an (optimal) ordered partition for a set of n objects, $S = \{O_1, \ldots, O_n\}$, defined by a collection of M mutually exclusive and exhaustive subsets of S, denoted $S_1, S_2, \ldots, S_M$, for which an order is imposed on the placement of the classes, $S_1 \prec S_2 \prec \cdots \prec S_M$, but not on their constituent objects. The data available to guide this search are assumed to be in the form of either (i) an $n \times n$ symmetric proximity matrix $\mathbf{P} = \{p_{ij}\}$, where p_{ij} $(= p_{ji} \geq 0$, and $p_{ii} = 0)$ is a dissimilarity for the objects O_i and O_j in which larger values indicate more dissimilar objects, or (ii) an $n \times n$ skew-symmetric proximity matrix $\mathbf{T} = \{t_{ij}\}$, where t_{ij} $(= -t_{ji}$, and $t_{ii} = 0)$ characterizes a dominance relation for the objects O_i and O_j (i.e., when $t_{ij} > 0$, O_i dominates O_j with the degree of dominance given by t_{ij}). In general, the identification of an optimal ordered partition for S will be carried out by the maximization of some index of merit that is intended to measure how well a given ordered partition reflects the data in $\mathbf{P}$ or $\mathbf{T}$.[1]

The organization of the paper is briefly as follows. Section 2 proposes several alternate measures of merit for indexing how well a particular ordered partition reflects the data in a given proximity matrix $\mathbf{P}$ or $\mathbf{T}$, and Section 3 presents an explicit optimization strategy based on dynamic programming (DP) that allows the

1991 *Mathematics Subject Classification.* Primary 62H30, 92G30; Secondary 90C39, 62-07.

[1] If the available data are originally in the form of an $n \times n$ non-symmetric proximity matrix, say $\mathbf{Q}$, the decomposition $\mathbf{Q} = (1/2)(\mathbf{Q} + \mathbf{Q}') + (1/2)(\mathbf{Q} - \mathbf{Q}')$ would provide symmetric and skew-symmetric matrices that could be analyzed separately.

identification of a (globally) optimal ordered partition of S maximizing whichever merit measure is chosen. The short subsection 3.1 discusses how the general DP formulation may also be extended heuristically to allow the analysis of object sets of a size beyond the computational limits imposed by a DP strategy that could guarantee global optimality. A numerical illustration is presented in Section 4 which relies on a computer program we have developed for carrying out the recursive DP process using any of the measures of merit introduced in Section 2. Finally, Section 5 presents two possible extensions of the task of constructing ordered partitions: the identification of a median for a collection of proximity matrices, and the construction of a sequence of ordered partitions and the generalization this may provide to the (ubiquitous) concept of an ultrametric central to the task of finding a hierarchical clustering for an object set S defined through a collection of unordered partitions.

2. Measures of Merit for an Ordered Partition

When data are in the form of either a symmetric matrix $\mathbf{P}$ or a skew-symmetric matrix $\mathbf{T}$, there are two general types of indices of merit that can be used to characterize the adequacy of a specific ordered partition in reflecting the pattern of proximities in $\mathbf{P}$ or $\mathbf{T}$. One is defined directly by how well an ordering of the proximities is mirrored by the ordering of the M classes, and the second is obtained indirectly through a secondary task of locating a set of (optimally) estimated coordinates for the M classes and how differences in the latter can reconstruct the proximities in a least-squares sense. We begin by assuming the availability of a symmetric proximity matrix $\mathbf{P}$ and develop these two types of merit indices in this context, and then proceed to do the same for a skew-symmetric matrix $\mathbf{T}$.

For a symmetric matrix $\mathbf{P}$·

The patterning of proximities in $\mathbf{P}$ will be considered reflected in the ordered partition, $S_1 \prec \cdots \prec S_M$, whenever an order for a pair of proximities implied by the ordered partition is actually present in $\mathbf{P}$. Explicitly, suppose we consider three distinct objects $O_{i'}, O_{k'}, O_{j'} \in S$ for which $O_{i'} \in S_i, O_{k'} \in S_k, O_{j'} \in S_j$, and $S_i \prec S_k \prec S_j$. If this latter class ordering is consistent with the data in $\mathbf{P}$, then the three proximities, $p_{i'k'}, p_{i'j'}$, and $p_{k'j'}$, should have the relation: $p_{i'k'} \leq p_{i'j'}$ and $p_{k'j'} \leq p_{i'j'}$ (or equivalently, $\max\{p_{i'k'}, p_{k'j'}\} \leq p_{i'j'}$); otherwise, if one or both of these inequalities fail, the order of the three classes would be inconsistent with the proximity data for these three objects.

An overall measure of merit will merely aggregate a function of the proximity differences for all distinct object triples contained within three classes of the ordered partition. Formally, we characterize the (direct) merit index as $\sum_{k=1}^{M} F(S_k)$, where $F(S_1) = F(S_M) \equiv 0$, and for $2 \leq k \leq M-1$, the contribution $F(S_k)$ for the k^{th}-placed class S_k is defined by a sum over all distinct object triples, $O_{i'}, O_{k'}, O_{j'} \in S$, where $O_{i'}$ belongs to a class that precedes S_k (i.e., $O_{i'} \in S_1 \cup \cdots \cup S_{k-1}$), $O_{k'} \in S_k$, and $O_{j'}$ belongs to a class that succeeds S_k (i.e., $O_{j'} \in S_{k+1} \cup \cdots \cup S_M$). Explicitly, $F(S_k) =$

$$\sum_{O_{i'} \in S_1 \cup \cdots \cup S_{k-1},\ O_{k'} \in S_k,\ O_{j'} \in S_{k+1} \cup \cdots \cup S_M} \{f(p_{i'k'}, p_{i'j'}) + f(p_{k'j'}, p_{i'j'})\},$$

for $f(\cdot, \cdot)$ defining a function of how the order relation present for its two arguments counts in the total merit measure. Letting x and y denote two proximities, we

explicitly consider two specifications for $f(x,y)$: $f(x,y) = \mid y - x \mid \operatorname{sign}(y-x)$ and $f(x,y) = \operatorname{sign}(y-x)$, where $\operatorname{sign}(y-x) = +1$ if $y - x > 0$; $= 0$ if $y - x = 0$; $= -1$ if $y - x < 0$. The use of these are referred to respectively as weighted and unweighted gradient measures of merit for the ordered partition, $S_1 \prec \cdots \prec S_M$.[2]

To obtain the more indirect measure of merit for an ordered partition, we first introduce a least-squares loss function

$$\sum_{k \le k'} \sum_{O_{i_k} \in S_k,\ O_{j_{k'}} \in S_{k'}} (p_{i_k j_{k'}} - \mid x_{k'} - x_k \mid)^2,$$

where the collection of coordinates is subject to the constraint that $x_1 \le x_2 \le \cdots \le x_M$ and $\sum_{k=1}^{M} n_k x_k = 0$. For a fixed ordered partition, an appropriate set of coordinates could be constructed through an inequality constrained least-squares procedure, such as in Dykstra [**B**], that would fit an $n \times n$ matrix $\mathbf{D} = \{d_{i_k j_{k'}}\}$ to $\mathbf{P} = \{p_{i_k j_{k'}}\}$, where the entries in $\mathbf{D}$ are subject to the inequality/equality constraints implied by the coordinate representation: $d_{i_k j_{k'}} = 0$ for $k = k'$; all $d_{i_k j_{k'}}$ are equal for $O_{i_k} \in S_k$ and $O_{j_{k'}} \in S_{k'}$; and $d_{i_k j_{k'}} + d_{j_{k'} h_{k''}} = d_{i_k h_{k''}}$ for $O_{i_k} \in S_k, O_{j_{k'}} \in S_{k'}$, and $O_{h_{k''}} \in S_{k''}$ when $S_k \prec S_{k'} \prec S_{k''}$. Given the fitted matrix $\mathbf{D}$, the nonnegative spacing between two adjacent classes, $S_k \prec S_{k+1}$, is given by the (common) value $d_{i_k j_{k+1}}$ for $O_{i_k} \in S_k$ and $O_{j_{k+1}} \in S_{k+1}$. The set of $n-1$ nonnegative adjacent spacings, in conjunction with the condition $\sum_{k=1}^{M} n_k x_k = 0$, allows an appropriate set of coordinates to be retrieved.

Because we wish to use the merit measure primarily in the search for an optimal ordered partition and not just to index a fixed one, it is very convenient that we can actually proceed without the explicit step of fitting a matrix $\mathbf{D}$. Specifically, the least-squares loss-function can be algebraically rewritten as

$$\sum_{i<j} p_{ij}^2 + n \sum_{k=1}^{M} n_k [x_k - (1/n) I(S_k)]^2 - (1/n) \sum_{k=1}^{M} n_k [I(S_k)]^2,$$

where

$$I(S_1) = -(1/n_1) \sum_{O_{k'} \in S_1,\ O_{j'} \in S - S_1} p_{k'j'},\quad I(S_M) = (1/n_M) \sum_{O_{k'} \in S_M,\ O_{i'} \in S - S_M} p_{k'i'},$$

and for $2 \le k \le M-1$,

$$I(S_k) = (1/n_k) \left(\sum_{O_{i'} \in S_1 \cup \cdots \cup S_{k-1},\ O_{k'} \in S_k} p_{k'i'} - \sum_{O_{j'} \in S_{k+1} \cup \cdots \cup S_M,\ O_{k'} \in S_k} p_{k'j'} \right).$$

An ordered partition that would minimize the least-squares loss function, say $S_1^* \prec \cdots \prec S_M^*$, can be shown to have the property that $(1/n) I(S_1^*) \le \cdots \le (1/n) I(S_M^*)$ and $\sum_{k=1}^{M} n_k (1/n) I(S_k^*) = 0$ (these inequalities can be proven by the observation that no adjacent pair of classes in an optimal ordered partition can be interchanged for a reduction in the loss criterion as long as the off-diagonal proximities in $\mathbf{P}$ are all positive); the value of the least-squares criterion for the minimizing ordered

[2]The index $F(S_k)$ includes both within-row and within-column comparisons in $\mathbf{P}$ (defined respectively though the functions $f(p_{i'k'}, p_{i'j'})$ and $f(p_{k'j'}, p_{i'j'})$); a variation on the use of the overall merit criterion would rely on only one of these.

partition is simply

$$\sum_{i<j} p_{ij}^2 - (1/n) \sum_{k=1}^{M} n_k [I(S_k^*)]^2.$$

Thus, we can define the optimal set of coordinates $x_k^* = (1/n) I(S_k^*)$ for $1 \leq k \leq M$ and concentrate on the actual search for an optimal ordered partition by maximizing the merit measure $\sum_{k=1}^{M} n_k [I(S_k)]^2$ over all possible ordered partitions of S.

For a skew-symmetric matrix $\mathbf{T}$:

The patterning of entries in $\mathbf{T}$ will be assumed reflected in an ordered partition whenever the dominance relation implied by the ordered classes is actually present in $\mathbf{T}$. Explicitly, for an (ordered) object pair $O_{i'} \in S_i$ and $O_{k'} \in S_k$ where $S_i \prec S_k$, and with objects in S_i dominating objects in S_k, consistency could be defined by the condition $t_{i'k'} \geq 0$. Thus, an overall measure of merit might merely sum those values in $\mathbf{T}$ for which the first object belongs to a lower-ordered class than the second. Formally we define the (direct) merit index as $\sum_{k=1}^{M} J(S_k)$, where $J(S_1) \equiv 0$, and for $2 \leq k \leq M$,

$$J(S_k) = \sum_{O_{i'} \in S_1 \cup \cdots \cup S_{k-1},\ O_{k'} \in S_k} t_{i'k'}.$$

This index is merely the sum of proximities in $\mathbf{T}$ from objects in S_k to $S_{k'}$ for all $k < k'$.

The generation of an indirect measure of merit for an ordered partition based on $\mathbf{T}$ and the estimation of a set of coordinates is approached in a slightly different way than for the case of a symmetric matrix $\mathbf{P}$ since closed-form solutions can always be given for the coordinate estimation phases. To be explicit, suppose $S_1, \ldots, S_M$ denote the classes of a fixed unordered partition and we wish to solve the least-squares task of finding a collection of optimal coordinates, $x_1, \ldots, x_M$ (not necessarily ordered), to minimize

$$(1/2) \sum_{k,k'} \sum_{O_{i_k} \in S_k,\ O_{j_{k'}} \in S_{k'}} (t_{i_k j_{k'}} - (x_k - x_{k'}))^2.$$

The solution is immediate by letting $x_k = (1/n) L(S_k)$, where for $1 \leq k \leq M$,

$$L(S_k) = (1/n_k) \sum_{O_{k'} \in S_k,\ O_{i'} \in S - S_k} t_{k'i'},$$

and $\sum_{k=1}^{M} n_k x_k = 0$; the classes can then be ordered according to their increasing coordinate values to identify an ordered partition based on the given classes $S_1, \ldots, S_M$. The least-squares criterion (assuming without loss of generality that $\sum_{k=1}^{M} n_k x_k = 0$) can be rewritten as

$$\sum_{i<j} t_{ij}^2 + n \sum_{k=1}^{M} n_k [x_k - (1/n) L(S_k)]^2 - (1/n) \sum_{k=1}^{M} n_k [L(S_k)]^2.$$

Thus, an optimal unordered partition can first be obtained by maximizing the merit index, $\sum_{k=1}^{M} n_k [L(S_k)]^2$, and the classes then ordered according to increasing coordinate values to identify an optimal ordered partition, $S_1^* \prec \cdots \prec S_M^*$.[3]

[3]The types of merit indices introduced in this section for ordered partitions have a long history in the literature for the one special case in which each class in the partition has only

Although we will not go into the details of implementation, we might note that it would also be possible to consider various weighted combinations of the merit measures introduced in this section to identify optimal ordered partitions. As one application, suppose (see footnote 1) that an originally given $n \times n$ non-symmetric proximity matrix $\mathbf{Q}$ is first decomposed into its symmetric and skew-symmetric component matrices. The use of a merit criterion defined as, say,

$$\sum_{k=1}^{M} n_k[I(S_k)]^2 + \sum_{k=1}^{M} n_k[L(S_k)]^2,$$

with the first term based on the symmetric matrix and the second on the skew-symmetric matrix, would be one method for identifying a single optimal ordered partition that would serve both the symmetric and skew-symmetric components of $\mathbf{Q}$.

3. Optimization by Dynamic Programming

Given $\mathbf{P}$ or $\mathbf{T}$ and one of the chosen measures of merit, the optimization task of locating an optimal ordered partition that maximizes merit can proceed through a recursive (dynamic programming) strategy. Explicitly, define a collection of M sets of entities, $\Omega_1, \ldots, \Omega_M$, where Ω_k contains all $2^n - 1$ nonempty subsets of the n object subscripts; let $\mathcal{F}(A_k)$ for $A_k \in \Omega_k$ be the optimal value for placing k classes at the first k order positions, where A_k is the union of these k classes, and define recursively

$$\mathcal{F}(A_k) = \max[\mathcal{F}(A_{k-1}) + M(A_{k-1}, A_k)].$$

Here, $A_{k-1} \in \Omega_{k-1}$, $A_k \in \Omega_k$, and the maximization is taken (exhaustively) over all proper subsets $A_{k-1} \subset A_k$. The increment in merit, $M(A_{k-1}, A_k)$, when placing the class $A_k - A_{k-1}$ at the k^{th} order position is obtained from one of the set functions given above: $F(A_k - A_{k-1}), n_k[I(A_k - A_{k-1})]^2, J(A_k - A_{k-1})$, or $n_k[L(A_k - A_{k-1})]^2$. Beginning with the direct computation of $\mathcal{F}(A_1)$ for $A_1 \in \Omega_1$, and proceeding with the recursive generation of $\mathcal{F}(A_k)$ for $k = 2, \ldots, n$, an optimal ordered partition is finally identified by $\mathcal{F}(A_M)$ for $S = A_M$; an ordered partition attaining this value can be constructed by working backwards through the recursive steps. In addition, optimal ordered partitions into 2 through $M - 1$ classes are identified by $\mathcal{F}(A_k)$ for $A_k = S \in \Omega_k$, $2 \leq k \leq M - 1$. (For convenience of reference, the program we use to implement the DP recursion will be referred to by the acronym HPOPARUN, for '*H*euristic *P*rogramming *O*rdered *Par*tition *Un*restricted', where the term 'heuristic'is included because of the included extension for larger object sets, as discussed in the subsection to follow.)[4]

---

a single object, i.e., when we seek an optimal ordering for the objects along a continuum. For extensive reviews of this one special case and characterizations of the kind of optimality criteria that can be satisfied, the reader is referred to Hubert [**C**] (for skew-symmetric matrices) and to Hubert and Arabie [**D**] (for symmetric matrices).

[4]If interests were centered *only* on optimal ordered partitions in which each class contained a *single* object (see footnote 3), a different set of entities, $\Omega_1, \ldots \Omega_n$, could be defined and over which the recursion could take place. Here, Ω_k would only have to contain all k member subsets of the n object subscripts, $1 \leq k \leq n$. The reader is referred to Hubert and Arabie [**D**] or Hubert and Golledge [**F**] for explicit discussions and implementations. In general, within the context of finding optimal ordered partitions into from 2 through n classes, the much larger sets Ω_k are necessary, containing *all* $2^n - 1$ nonempty subsets of the n object subscripts.

3.1. Heuristic Extensions for Larger Object Sets. The storage requirements of random access memory (RAM) necessary for a dynamic programming approach to the construction of optimal ordered partitions can become quite enormous when the number of objects in S is even moderate in size. Required for carrying out the proposed recursive strategy is the availability of large arrays associated with the sets, $\Omega_1, \ldots, \Omega_M$, that contain for all $2^n - 1$ nonempty subsets of S the optimal recursively-constructed values $\mathcal{F}(A_k)$ for $A_k \in \Omega_k$, as well as a mechanism for keeping track of what subset was placed at the k^{th} order position that led to these optimal values. Given the usual Pentium-level processors now commonly available and the amount of RAM these systems usually contain, the program we have developed can deal (optimally) with object set sizes of about 20, but requires the capability of Fortran90 to allocate very large arrays dynamically (and inform the user whether sufficient RAM exists on the system to solve the problem of the size being requested.) For object sets larger than this limit, HPOPARUN allows the option of finding optimal ordered partitions when the basic objects to be partitioned into ordered classes are themselves subsets of S. By the judicious and repeated use of this option, we have been able to approach object sets that are fairly large in size (e.g., in the low hundreds).

The analysis strategy we suggest is to begin by identifying a partition of S (possibly through heuristic means, such as from a complete-link hierarchical clustering truncated at the desired number of classes), and treat the classes of this partition as the basic units to be grouped into ordered classes. Given this latter (initial) collection of ordered classes, various aggregates of adjacently-placed classes can then be united and treated as the units to be partitioned but allowing the objects in some of the classes to be considered individually. This process can be repeated until no change occurs for any chosen aggregation of adjacently-placed objects (that are being considered the units to be partitioned into ordered classes). Obviously, an absolute guarantee of optimality is not possible through this type of heuristic search, but the eventual stability achieved leads to an ordered partition that is usually very good although not verifiably optimal.

4. A Numerical Illustration

The data set that we use to illustrate the construction of ordered partitions (and relying on the program HPOPARUN) is a very old one, originally collected in 1929 for a study of the influence of motion pictures on children's attitudes (see Thurstone [**H**], pp. 309–319). Both before and after seeing a film entitled *Street of Chance*, which depicted the life of a gambler, 240 school children were asked to compare the relative seriousness of thirteen offenses presented in all 78 possible pairs: bankrobber, gambler, pickpocket, drunkard, quack doctor, bootlegger, beggar, gangster, tramp, speeder, petty thief, kidnapper, and smuggler. The original data are given in Table 1, where each entry shows the proportion of children who rated the offense listed in the column to be more serious than the offense listed in the row. The above-diagonal entries were obtained before the showing of the film; those below the main diagonal were collected after. The obvious substantive question here involves the effect of the film on the assessment of the offense of being a gambler.

TABLE 1. The proportions of school children who evaluate the column offense as more serious than the row offense (taken from Thurstone [**H**], p. 311). The above-diagonal entries are before showing the film *Street of Chance*; those below the diagonal were collected after viewing the motion picture. Note: Apparently, the single pair (petty thief, kidnapper) was inadvertently not presented for evaluation, although no mention of this anomaly is made in Thurstone [**H**]. The values of .98 for (a) and .03 for (b) will be used based on an assumption of strong stochastic transitivity (e.g., see Bezembinder and van Acker [**A**]) and the observed proportions for the two pairs (petty thief, bankrobber) and (kidnapper, bankrobber).

offense	1	2	3	4	5	6	7	8	9	10	11	12	13
1:bankrobber	—	.07	.08	.05	.27	.29	.01	.50	.00	.06	.02	.73	.21
2:gambler	.79	—	.71	.52	.76	.92	.07	.92	.05	.41	.49	.90	.81
3:pickpocket	.93	.51	—	.25	.67	.75	.02	.86	.02	.39	.42	.87	.68
4:drunkard	.95	.70	.70	—	.81	.95	.01	.92	.03	.37	.62	.91	.87
5:quack doctor	.67	.36	.28	.16	—	.49	.02	.70	.02	.12	.22	.64	.55
6:bootlegger	.70	.31	.30	.13	.50	—	.00	.79	.01	.09	.26	.68	.50
7:beggar	.98	.95	.97	.94	.98	.98	—	.96	.42	.86	.96	1.0	.99
8:gangster	.50	.18	.13	.11	.32	.27	.01	—	.02	.08	.08	.36	.31
9:tramp	1.0	.96	.98	.96	.99	.98	.64	.99	—	.91	.97	.99	1.0
10:speeder	.94	.73	.68	.67	.89	.90	.21	.94	.13	—	.58	.90	.92
11:petty thief	.97	.64	.62	.47	.81	.76	.06	.89	.05	.36	—	a	.78
12:kidnapper	.38	.27	.16	.08	.35	.30	.02	.62	.01	.08	b	—	.27
13:smuggler	.73	.31	.30	.16	.46	.49	.02	.66	.02	.11	.24	.64	—

The data of Table 1 can be used to generate both symmetric and skew-symmetric (before and after) matrices that can illustrate the construction of ordered partitions. Explicitly, Table 2 provides symmetric before and after proximity matrices with entries for each pair of offenses defined as the absolute values of the difference in the proportions of rating one offense more serious than the other. For example, since the proportion judging a bankrobber more serious than a bootlegger is .71 before the movie was shown, a symmetric dissimilarity of $.42 = |\ .71 - .29\ |$ is given for the pair (bootlegger, bankrobber) in the corresponding proximity matrix. The two proximity matrices so constructed are given in the upper- and lower-triangular portions of Table 2 (but where for graphical convenience, the rows and columns have been reordered to correspond to optimal sequencings described below). In addition to the symmetric matrices represented in Table 2, skew-symmetric variants that indicate a directionality of dominance based on the differences in proportions of rating one offense more serious than a second are indicated by an asterisk (*), reflecting that this entry was negative before taking an absolute value. For example, for the before matrix entry of .46* associated with the pair (kidnapper(12), bankrobber(1)), the proportion judging bankrobber(1) more serious than kidnapper(12) is .27; thus the entry in Table 2 is the absolute value of $.27 - .73 = -.46$, but the negative sign shows the greater preceived seriousness of being a kidnapper (and contrary to the ordering implicit in Table 2 from least to greatest in seriousness).

TABLE 2. Symmetric dissimilarity matrices constructed for thirteen offenses using the absolute values of the skew-symmetric proximities constructed from the entries in Table 1. The above-diagonal entries are before showing the film *Street of Chance*; those below the diagonal were after viewing the motion picture. Note: Entries with an asterisk (*) were negative before taking the absolute value.

offense	9	7	10	4	2	11	3	5	13	6	12	8	1	
9:tramp	—	.16	.82	.94	.90	.94	.96	.96	1.0	.98	.98	.97	1.0	9
7:beggar	.28	—	.72	.98	.86	.92	.96	.96	.98	1.0	1.0	.92	.98	7
10:speeder	.74	.58	—	.26	.18	.16	.22	.76	.84	.82	.80	.84	.88	10
4:drunkard	.92	.88	.34	—	.04*	.24	.50	.62	.74	.90	.82	.84	.90	4
11:petty thief	.90	.88	.28	.06	—	.02*	.42	.52	.62	.84	.80	.84	.86	2
3:pickpocket	.96	.94	.36	.40	.24	—	.16	.56	.56	.48	.96	.84	.96	11
2:gambler	.92	.90	.46	.40	.28	.02	—	.34	.36	.50	.74	.72	.84	3
6:bootlegger	.96	.96	.80	.74	.52	.40	.38	—	.10	.02*	.28	.40	.46	5
13:smuggler	.96	.96	.78	.68	.52	.40	.38	.02	–	.00	.46	.38	.58	13
5:quack doctor	.98	.96	.78	.68	.62	.44	.28	.00	.08*	—	.36	.58	.42	6
12:kidnapper	.98	.96	.84	.84	.94	.68	.46	.40	.28	.30	—	.28	.46*	12
8:gangster	.98	.98	.88	.78	.78	.74	.64	.46	.32	.36	.24	—	.00	8
1:bankrobber	1.0	.96	.88	.90	.94	.86	.58	.40	.46	.34	.24*	.00	—	1
	9	7	10	4	11	3	2	6	13	5	12	8	1	

For some brevity, we will present for the various measures of matrix patterning the original ordering using single objects (i.e., ordered partitions with thirteen classes) and an optimal ordered partition with five classes. In all cases, there was a very precipitous change in the merit measures when moving from five to four classes, and therefore, the choice of presenting only those optimal ordered partitions with five classes is not arbitrary. As shown in the results summarized below, the five ordered (according to increasing severity) classes of offenses consistently include the offense of being a gambler(2) in the second class before viewing the film, and in the third class after (we might also comment that in most of the analyses reported, the 5-class and the 13-class ordered partitions are completely consistent in the sense that the classes in the former are defined by consecutively-placed single objects in the later; the exceptions are for the gradient measures and a few adjacently-located objects). Given the general consistency in the orderings and the objects placed in the ordered classes using the before and after matrices except for 'gambler', the showing of the film apparently had the effect of changing the school children's perception of its seriousness, and in the direction of evaluating it as a much more serious offense after seeing the motion picture than before.

Before viewing:
symmetric proximity; coordinate representation

object order	9	7	10	4	2	11	3	5	13	6	12	8	1
coordinates	−.82	−.78	−.33	−.26	−.23	−.17	−.02	.27	.29	.32	.50	.59	.64
(5 classes):	{ −.80 }		{	−.25		}	{−.02}	{	.29	}	{	.58	}

residual sum-of-squares: (13 classes) 3.307; (5 classes) 3.635

symmetric proximity; unweighted gradient
object order 9 7 4 10 2 11 3 5 13 6 12 8 1
(5 classes) { } { } { } {13 6 8} {12 1}
index of gradient comparisons: (13 classes) 431; (5 classes) 289

symmetric proximity; weighted gradient
object order 9 7 10 4 2 11 3 5 13 6 12 8 1
(5 classes) { } { } { } {6 8} {12 1}
index of gradient comparisons: (13 classes) 161.45; (5 classes) 100.08

skew-symmetric proximity; above-diagonal sum
object order 9 7 10 2 4 11 3 6 5 13 1 12 8
(5 classes) { } { } { } { } { }
above-diagonal sum: (13 classes) 49.93; (5 classes) 48.27

skew-symmetric proximity; coordinate representation

object order	9	7	10	4	2	11	3	5	13	6	1	12	8
coordinates	−.82	−.78	−.33	−.26	−.23	−.18	−.02	.27	.29	.32	.57	.57	.59
(5 classes)	{	−.80	}	{	−.25	}	{−.02}	{	.29	}	{	.58	}

residual sum-of-squares: (13 classes) 3.457; (5 classes) 3.635

After viewing:
symmetric proximity; coordinate representation

object order	9	7	10	4	11	3	2	6	13	5	12	8	1
coordinates	−.81	−.75	−.39	−.26	−.21	−.05	−.02	.27	.27	.29	.48	.55	.58
(5 classes)	{	−.78	}	{	−.29	}	{−.02}	{	.28	}	{	.54	}

residual sum-of-squares: (13 classes) 2.302; (5 classes) 2.674

symmetric proximity; unweighted gradient
object order 9 7 10 4 11 3 2 13 6 5 12 8 1
(5 classes) { } { } { } { } { }
index of gradient comparisons: (13 classes) 491; (5 classes) 331

symmetric proximity; weighted gradient
object order 9 7 10 4 11 3 2 6 13 5 12 8 1
(5 classes) {9 7 4} {10 11 3} { } {5 8} {12 1}
index of gradient comparisons: (13 classes) 165.08; (5 classes) 101.84

skew-symmetric proximity; above-diagonal sum
object order 9 7 10 4 11 3 2 5 6 13 1 12 8
(5 classes) { } { } { } { } { }
above-diagonal sum: (13 classes) 47.42; (5 classes) 45.86

skew-symmetric proximity; coordinate representation

object order	9	7	10	4	11	3	2	6	5	13	12	1	8
coordinates	−.81	−.75	−.39	−.26	−.21	−.05	.02	.27	.28	.29	.51	.55	.55
(5 classes)	{	−.78	}	{	−.29	}	{	−.02	}	{	.28	}	{	.54	}

residual sum-of-squares: (13 classes) 2.369; (5 classes) 2.674

5. Some Extensions and Generalizations

There are a variety of topics related to the construction of optimal ordered partitions that could be pursued and linked to problems discussed at length in the

classification literature. We will mention only two such possibilities briefly in closing, with the first concerned with constructing a 'median'matrix for a collection of proximity matrices that contains dichotomous (0/1) entries representing an ordered partition (which closely follows the work of Mirkin, e.g., see [**G**], pp. 90-116), and the second suggesting a way of extending ordered partitions to a hierarchical framework as a way of generalizing the notion of an ultrametric. Suppose a collection of N $n \times n$ (nonnegative and possibly nonsymmetric) proximity matrices for an object set S is available that we denote by $\mathbf{R}^{(1)}, \ldots, \mathbf{R}^{(N)}$, where $\mathbf{R}^{(k)} = \{r_{ij}^{(k)}\}$ and $0 \leq r_{ij}^{(k)} \leq 1$ for $1 \leq i, j \leq n$ and $1 \leq k \leq N$. The optimization task we pose is to find an optimal median matrix $\mathbf{R}$ that contains (strictly) 0 or 1 entries representing an ordered partition $S_1 \prec \cdots \prec S_M$, i.e., if $O_i \in S_k, O_j \in S_{k'}$, and $S_k \prec S_{k'}$, then $r_{ij} = 1$, otherwise $r_{ij} = 0$. Explicitly, we seek $\mathbf{R}$ to minimize $\sum_{k=1}^{N} d(\mathbf{R}^{(k)}, \mathbf{R})$, where $d(\mathbf{R}^{(k)}, \mathbf{R}) = \sum_{i,j} \mid r_{ij}^{(k)} - r_{ij} \mid$. Following the type of reductions given in Mirkin [**G**], pp. 90-116,

$$\sum_{k=1}^{N} d(\mathbf{R}^{(k)}, \mathbf{R}) = \sum_{k=1}^{N} \sum_{i,j} r_{ij}^{(k)} - 2 \sum_{i,j} (\sum_{k=1}^{N} r_{ij}^{(k)} - \frac{N}{2}) r_{ij},$$

and thus minimizing $\sum_{k=1}^{N} d(\mathbf{R}^{(k)}, \mathbf{R})$ is equivalent to maximizing $\sum_{i,j} (\sum_{k=1}^{N} r_{ij}^{(k)} - \frac{N}{2}) r_{ij}$. Because $\{\sum_{k=1}^{N} r_{ij}^{(k)} - \frac{N}{2}\}$ can be treated as a proximity matrix, and given the form of the to-be-identified $\mathbf{R}$, an optimal median proximity matrix can be solved using the index of merit given earlier in the form $\sum_{k=1}^{M} J(S_k)$ (although discussed explicitly for use with a skew-symmetric matrix $\mathbf{T}$, this particular merit measure generalizes immediately to the use of any proximity matrix). Thus, the task of finding an optimal median matrix $\mathbf{R}$ that represents M ordered classes reduces to identifying an optimal ordered partition containing M classes based on a proximity matrix defined by an aggregation over the initial N proximity matrices, $\mathbf{R}^{(1)}, \ldots, \mathbf{R}^{(N)}$, and one of the indices of merit used previously.

In our discussion of constructing optimal ordered partitions, possibly for differing numbers of classes, no particular assumption was made about the relationship of one ordered partition to another. Suppose, however, that we proceed to construct for S a collection of T ordered partitions into anywhere from 1 to n classes, where each class in a partition defines a consecutive set of objects with respect to some fixed ordering of the n objects. For convenience, we denote the T partitions as $\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_T$, where $\mathcal{P}_1$ is a partition containing n classes, $\mathcal{P}_T$ includes only a single class, and $\mathcal{P}_{t-1}$ has more classes than $\mathcal{P}_t$ for $t \geq 2$ (and as noted, each class in any partition contains objects consecutive in some common ordering).

Based on $\mathcal{P}_1, \ldots, \mathcal{P}_T$, if a corresponding collection of $n \times n$ 0/1 dissimilarity matrices $\mathbf{P}_1, \ldots, \mathbf{P}_T$ is constructed, where a 0 in $\mathbf{P}_t$ indicates an object pair defined within a class in $\mathcal{P}_t$, and 1 otherwise, then for any collection of nonnegative weights $\alpha_1, \alpha_2, \ldots, \alpha_T$, the dissimilarity matrix, say, $\mathbf{P}_\alpha = \{p_{ij}^{(\alpha)}\} \equiv \sum_{t=1}^{T} \alpha_t \mathbf{P}_t$, defines a metric on the objects (based on the observation that sums of metrics are metric [but with the possible extension that allows some dissimilarities to be zero for nonidentical objects]). Depending on the constraints placed on $\mathcal{P}_1, \ldots, \mathcal{P}_T$, more restrictive forms for the metric defined by $\mathbf{P}_\alpha$ ensue; and specific to the restrictions made, it may be possible to retrieve $\mathcal{P}_1, \ldots, \mathcal{P}_T$ and $\alpha_1, \ldots, \alpha_T$ given only $\mathbf{P}_\alpha$, provide convenient graphical representations for the collection $\mathcal{P}_1, \ldots, \mathcal{P}_T$, or somehow

to approach the task of constructing $\mathcal{P}_1, \ldots, \mathcal{P}_T$ and $\alpha_1, \ldots, \alpha_T$ from some given proximity matrix $\mathbf{P}$ so that $\mathbf{P}_\alpha$ approximates $\mathbf{P}$ in some explicitly defined sense. The obvious prime exemplar for this type of structure would be when $\mathcal{P}_t$ is formed from $\mathcal{P}_{t-1}$ by uniting two or more classes in the latter. The entries in $\mathbf{P}_\alpha$ then satisfy the ultrametric inequality ($p_{ij}^{(\alpha)} \leq \max\{p_{ik}^{(\alpha)}, p_{jk}^{(\alpha)}\}$ for all $O_i, O_j, O_k \in S$), the partition hierarchy and the weights are retrievable given only $\mathbf{P}_\alpha$, and a representation of the hierarchical clustering can be given in the form of what is usually called a dendrogram.

TABLE 3. For the 'after' proximity matrix given in Table 2, the upper-triangular portion provides an (optimal) least-squares ultrametric; the lower-triangular portion provides a least-squares fitted structure based on ordered partitions.

offense	9	7	10	4	11	3	2	6	5	13	12	8	1	
9:tramp	—	.28	.92	.92	.92	.92	.92	.92	.92	.92	.92	.92	.92	9
7:beggar	.28	—	.92	.92	.92	.92	.92	.92	.92	.92	.92	.92	.92	7
10:speeder	.62	.62	—	.31	.31	.36	.36	.67	.67	.67	.67	.67	.67	10
4:drunkard	.62	.62	.31	—	.06	.36	.36	.67	.67	.67	.67	.67	.67	4
11:petty thief	.62	.62	.31	.06	—	.36	.36	.67	.67	.67	.67	.67	.67	11
3:pickpocket	.86	.86	.36	.36	.36	—	.02	.67	.67	.67	.67	.67	.67	3
2:gambler	.86	.86	.36	.36	.36	.02	—	.67	.67	.67	.67	.67	.67	2
6:bootlegger	.86	.86	.86	.86	.86	.62	.62	—	.02	.04	.37	.37	.37	6
13:smuggler	.86	.86	.86	.86	.86	.62	.62	.02	–	.04	.37	.37	.37	5
5:quack doctor	.86	.86	.86	.86	.86	.62	.62	.04	.04	—	.37	.37	.37	13
12:kidnapper	.86	.86	.86	.86	.86	.62	.62	.37	.37	.37	—	.24	.24	12
8:gangster	.86	.86	.86	.86	.86	.62	.62	.37	.37	.37	.24	—	.00	8
1:bankrobber	.86	.86	.86	.86	.86	.62	.62	.37	.37	.37	.24	.00	—	1
	9	7	10	4	11	3	2	6	13	5	12	8	1	

In the more general context where $\mathcal{P}_1, \ldots, \mathcal{P}_T$ are merely restricted to be ordered partitions, each defined by classes contiguous with respect to some given ordering for the objects in S, the entries in the matrix $\mathbf{P}_\alpha$ satisfy (at the least) the anti-Robinson condition (i.e., if $O_i \prec O_j \prec O_k$, then $p_{ik}^{(\alpha)} \geq \max\{p_{ij}^{(\alpha)}, p_{jk}^{(\alpha)}\}$), and can be constructed by sums of subsets of a collection of nonnegative weights $\alpha_1, \ldots, \alpha_T$, just as in the more restrictive ultrametric context. Thus, although the same number of parameters are needed to construct $\mathbf{P}_\alpha$ as for an ultrametric, the structures definable through ordered partitions restricted only by the class contiguity constraint are broader than those possible through the concept of an ultrametric. As an illustration of this greater generality, and thus, of the possibility of explaining more completely the structure of a given proximity matrix, we provide below a brief illustration using the 'after' matrix among the thirteen offenses given in the lower-triangular portion of Table 2. A (presumably) least-squares optimal ultrametric using the heuristic iterative projection strategy of Hubert and Arabie [**E**] is given in the upper-triangular portion of Table 3 with a 'variance-accounted-for' (vaf) measure of 81.12%. The lower-triangular portion of Table 3 provides an alternative structure (again fitted through least-squares) but with a larger vaf of 84.65%, corresponding to a collection of ordered partitions with classes contiguous

in the object order used to present the rows of Table 3 (the ordered partitions were initially identified using the merit criterion based on coordinate estimation, i.e., $\sum_{k=1}^{M} n_k[I(S_k)]^2$, for $M = 2, \ldots, 13$). We present below a set of estimated weights and the ordered partitions that induce the upper- and lower-triangular portions of Table 3.

Classes α	Ordered partition
For the optimal fitted ultrametric:	
13	.00 9 7 10 4 11 3 2 6 5 13 12 8 1
12	.02 9 7 10 4 11 3 2 6 5 13 12 (8 1)
11	.00 9 7 10 4 11 (3 2) 6 5 13 12 (8 1)
10	.02 9 7 10 4 11 (3 2) (6 5) 13 12 (8 1)
9	.02 9 7 10 4 11 (3 2) (6 5 13) 12 (8 1)
8	.18 9 7 10 (4 11) (3 2) (6 5 13) 12 (8 1)
7	.04 9 7 10 (4 11) (3 2) (6 5 13) (12 8 1)
6	.03 (9 7) 10 (4 11) (3 2) (6 5 13) (12 8 1)
5	.05 (9 7) (10 4 11) (3 2) (6 5 13) (12 8 1)
4	.01 (9 7) (10 4 11 3 2) (6 5 13) (12 8 1)
3	.30 (9 7) (10 4 11 3 2) (6 5 13 12 8 1)
2	.25 (9 7) (10 4 11 3 2 6 5 13 12 8 1)
1	— (9 7 10 4 11 3 2 6 5 13 12 8 1)
For the fitted structure based on ordered partitions:	
13	.00 9 7 10 4 11 3 2 6 13 5 12 8 1
12	.02 9 7 10 4 11 3 2 6 13 5 12 (8 1)
11	.00 9 7 10 4 11 (3 2) 6 13 5 12 (8 1)
10	.02 9 7 10 4 11 (3 2) (6 13) 5 12 (8 1)
9	.02 9 7 10 4 11 (3 2) (6 13 5) 12 (8 1)
8	.18 9 7 10 (4 11) (3 2) (6 13 5) 12 (8 1)
7	.04 9 7 10 (4 11) (3 2) (6 13 5) (12 8 1)
6	.03 (9 7) 10 (4 11) (3 2) (6 13 5) (12 8 1)
5	.05 (9 7) (10 4 11) (3 2) (6 13 5) (12 8 1)
4	.01 (9 7) (10 4 11 3 2) (6 13 5) (12 8 1)
3	.25 (9 7) (10 4 11 3 2) (6 13 5 12 8 1)
2	.24 (9 7 10 4 11) (3 2 6 13 5 12 8 1)
1	— (9 7 10 4 11 3 2 6 13 5 12 8 1)

References

[A] T. Bezembinder & P. van Acker, *Intransitivity in individual and social choice*, Similarity and Choice (E.D. Lantermann & H. Feger, eds.), Huber, Bern, 1980 (pp. 208–233).

[B] R. L. Dykstra, *An algorithm for restricted least squares regression*, Journal of the American Statistical Association, **78** (1983), 839–842.

[C] L. J. Hubert, *Seriation using asymmetric proximity measures*, British Journal of Mathematical and Statistical Psychology, **29**, 32–52.

[D] L. J. Hubert & P. Arabie, *Unidimensional scaling and combinatorial optimization*, Multidimensional Data Analysis (J. De Leeuw, W. Heiser, J. Meulman, & F. Critchley, eds.), DSWO Press, Leiden, 1986 (pp. 181–196).

[E] L. J. Hubert & P. Arabie, *Iterative projection strategies for the least-squares fitting of tree structures to proximity data*, British Journal of Mathematical and Statistical Psychology, **48** (1995), 281–317.

[F] L. J. Hubert & R. G. Golledge, *Matrix reorganization and dynamic programming: Applications to paired comparisons and unidimensional seriation*, Psychometrika, **46**, 429–441.
[G] B. G. Mirkin, *Group Choice*. Washington, DC, V. H. Winston, 1979.
[H] L. L. Thurstone, *The Measurement of Values*. The University of Chicago Press, Chicago, 1959.

Department of Psychology, 603 East Daniel Street, The University of Illinois, Champaign, Illinois 61820, USA.
E-mail address: l-hubert@uiuc.edu

Faculty of Management, Rutgers University, Newark, New Jersey, USA.
E-mail address: arabie@andromeda.rutgers.edu

Department of Data Theory, Leiden University, The Netherlands.
E-mail address: meulman@rulfsw.fsw.leidenuniv.nl

This page intentionally left blank

https://doi.org/10.1090/dimacs/037/22

DIMACS Series in Discrete Mathematics
and Theoretical Computer Science
Volume **37**, 1997

Multiple Trees: Fitting Two or More Tree Structures to Proximity Data

J. Douglas Carroll and Geert De Soete

ABSTRACT. In this paper representations of proximity data by multiple tree structure models are discussed. In such a representation, observed dissimilarities are approximated by a sum of two or more ultrametrics or by a sum of two or more path length metrics. Methods for fitting such tree structures to proximity data are described and the relationship to overlapping clustering based on the ADCLUS model is discussed. An approach is presented for organizing such overlapping cluster structures into single or multiple tree structures, utilizing a graph called the "nesting graph", whose vertices correspond to the clusters. Cliques (maximal complete subgraphs) of this graph correspond to maximal tree structures (tree structures that are not subtrees of any larger tree defined by a subset of the overlapping clusters) defined on the cluster structure. This approach, it is pointed out, can be applied to any overlapping cluster structure—not only ones based on the ADCLUS (or INDCLUS) model. The various models are applied for illustrative purposes to some data sets from the field of psychology.

1. Introduction

This paper defines, describes, and argues for the validity of *multiple* tree structure models for proximity data, in which *dis*similarities are modeled, for example, by distances that are sums of two, three, or more ultrametrics, each associated with a hierarchical tree structure, or by a sum of path length or additive metrics, each associated with an unrooted, or free tree, with a path length or additive distance defined between pairs of nodes representing objects. While these multiple tree structure models were developed originally with applications to cognitive and perceptual psychology in mind, so that the examples used to illustrate them are from the field of psychology, it is argued that such generalized hierarchical models also have many potential applications in biology (as well as in many other fields).

We describe methods for fitting multiple tree structures directly to proximity data, as well as an approach that uses the results of some form of overlapping clustering, such as one assuming the Shepard and Arabie [**32**] ADCLUS model, using the Arabie and Carroll [**1**] MAPCLUS method, INDCLUS [**5**] or SINDCLUS [**12**], for fitting this model. In a second approach we define a graph, called the "nesting graph," the vertices of which correspond to the clusters in the overlapping cluster structure. It is shown that a clique (or maximal complete subgraph) of the nesting graph corresponds to a tree structure, so that a "minimum clique covering"

1991 *Mathematics Subject Classification.* Primary 62H30; Secondary 92G30.

(a vertex clique covering with the smallest possible number of cliques) corresponds to a most parsimonious multiple tree structure reproducing the overlapping cluster structure.

Applications of these methods to fit multiple trees to data of Rosenberg and Kim [**30**], entailing dissimilarities among kinship terms derived from a similarity sorting task, as well as to some data on judged similarities of terms denoting types of interpersonal relationships due to Wish, Deutsch, and Kaplan [**34**], are described and discussed.

2. Single Tree Structure Models

A widely used method for fitting single tree structures for discrete representation of similarity (or other proximity) data is hierarchical clustering (e.g, [**22, 26**]). Hierarchical clustering yields a family of clusters such that any two clusters are disjoint or one includes the other. In the usual representation, the objects being clustered appear as terminal nodes of a tree and the distances between objects are the heights of the internal node defining the meeting point often called the "lowest common ancestor" node. The heights must be consistent with the partial ordering of the clusters defined by the hierarchical structure of the tree, i.e., for two clusters related by subset inclusion, the height associated with the larger cluster must be at least as large as the height associated with the smaller cluster. The model implies that, given two disjoint clusters, all distances between objects in the same cluster are no larger than distances between objects in the two different clusters, and that these between-cluster distances are equal (so that the resulting triangle is *acute isosceles* or *equilateral*). This property is equivalent to the ultrametric inequality (u.i.) and the tree representation is called an *ultrametric tree.* The ultrametric inequality states that

$$d_{ij} \leq \max(d_{ik}, d_{jk}) \tag{2.1}$$

for all i, j, k, or equivalently,

$$d_{ij} \leq d_{ik} = d_{jk}$$

for some ordering of the three points i, j, and k. Given a set of distances satisfying the ultrametric inequality the associated tree can easily be constructed and height values defined. Given a tree, an infinite family of ultrametrics can be defined. Once the height values are specified, however, the particular ultrametric is uniquely specified.

Several authors ([**22, 25, 26**]) independently suggested treating the problem of hierarchical clustering as one of fitting a certain geometric model, namely a rooted tree structure on which an ultrametric is defined, to proximity data. Hartigan proposed an explicit algorithm using combinatorial optimization techniques, that was aimed at optimizing a least squares criterion of fit between data "distances" and distances calculated from an ultrametric tree structure.

A few years later Carroll and Chang [**7**] devised a procedure that generalized Hartigan's approach in two different ways. While Hartigan's procedure, and essentially all other hierarchical clustering procedures, restricted the objects to *terminal* nodes of the tree, the Carroll and Chang procedure allowed some or all of the interior, or nonterminal nodes, to correspond to objects or stimuli. Secondly, they allowed a more flexible definition of the metric defined on the tree. In addition

to the ultrametric, two other kinds of metric were allowed. The first of these is a path length (or additive) metric, in which lengths are associated with branches, or links in the tree, and distance is simply the length of the (unique) path joining those two nodes of the tree. The second is a mixed case, in which "heights" are associated with nonterminal nodes *and* lengths with branches, while distance is defined as a sum of the path length and height of the lowest common ancestor node. This last "mixed" metric can be meaningfully distinguished from the simpler path length or additive metric *only* when some of the objects are represented by nonterminal nodes. When all objects are at terminal nodes, the ultrametric and the "mixed" metric are special cases of the path length metric. When some objects are at interior nodes the three models are all meaningfully different.

A tree with path length metric, or simply a path length tree is synonymous with the term "free tree" ([**14**]). Farris [**20**] calls this a tree with "four point metric," while Sattath and Tversky [**31**] call it an "additive similarity tree." Unlike an ultrametric tree which has a natural "root" node, a path length tree has no unique root. It is not necessary to think of it as being organized into a hierarchy. Both Cunningham [**14**] and Sattath and Tversky [**31**] have developed algorithms for fitting path length trees to data. Sattath and Tversky's method is a kind of natural generalization of the "pair group" method (e.g., single, average and complete linkage) often used to generate hierarchical clustering solutions. Cunningham's solution assumes that a four point condition which must be satisfied for path length distances, also holds, approximately, in the data. The four point condition states that the two largest sums of pairs of distances involving four objects must be equal; i.e., for some ordering of the four points i, j, k, and l, the following condition holds:

$$d_{ij} + d_{kl} \leq d_{ik} + d_{jl} = d_{jk} + d_{il}. \tag{2.2}$$

Cunningham's procedure tends to break down seriously, however, with even mildly "noisy" data.

3. Fitting Trees by Mathematical Programming Techniques

3.1. Fitting of a Single Ultrametric Tree. Carroll and Pruzansky [**10, 11**] (also see Carroll [**4**]) formulated a "mathematical programming" approach to fitting an ultrametric tree to proximity data. Basically, this approach attempts to find a least squares fit of a distance matrix constrained to satisfy the ultrametric inequality, by use of a "penalty function" which measures the degree of violation of that inequality, as defined in (2.1), to a given matrix of dissimilarities. The Carroll and Pruzansky approach to the OLS fitting of an ultrametric tree was improved by De Soete [**16**], and extended to the case of incomplete proximity data by De Soete [**17**]. This approach can be extended easily to the fitting of path length trees satisfying the four point condition in an indirect way which will be described later. A more direct procedure entailing a direct generalization of the Carroll and Pruzansky penalty function approach, using a penalty function to enforce the four point condition was proposed and implemented by De Soete [**15**], and extended to the missing data case by De Soete [**18**].

3.2. Fitting Multiple Ultrametric Tree Structures via Mathematical Programming Combined with Alternating Least Squares (ALS). There are many sets of proximity data that are not well represented by either nonoverlapping clusterings (or partitionings) or hierarchical clusterings. One alternative

model is the ADCLUS overlapping clustering model proposed by Shepard and Arabie [**32**] in which proximity data are assumed to arise from discrete attributes that define overlapping but nonhierarchically organized sets. It may be, however, that the attributes can be organized into two or more separate hierarchies. Each of these separate hierarchies could represent an organized family of subordinate and superordinate concepts. For example, in the case of animal names one might imagine one hierarchical conceptual scheme based on the phylogenetic scale, and another based on function or relationship to man, such as, tame versus wild. Tame animals could be further broken down into house pets versus outdoor pets, and so on. Two such conceptual hierarchical structures would obviously be far from independent of one another; whether or not an animal is a pet, for example, is not independent of the phylogenetic classification of the animal, but the structures could be sufficiently distinct that an appropriate technique could pull them apart. Such multiple hierarchies in data may often be obscured in standard clustering analyses, simply because of a possible high degree of correlation among separate structures. Another possible example of a two tree representation, relevant to the fitting of evolutionary tree structures, in which one tree represents phenetic structure and the other cladistic structure of a group of organisms. What is needed in such cases is a method for fitting a model entailing *multiple* ultrametric tree structures to data—a kind of multidimensional generalization of single ultrametric tree. Carroll and Pruzansky [**10, 11**] proposed a procedure for fitting such multiple ultrametric tree structures to a proximity data matrix, described below.

Consider fitting $\mathbf{\Delta}$, the data matrix (assumed hereafter to be a matrix of *ratio scale dis*similarities, which can be viewed as a matrix of approximate *distances*), via a mixture of hierarchical tree structures (HTS's), each of which will be assumed to satisfy the ultrametric inequality. In particular, we want to approximate Δ as a sum

$$\mathbf{\Delta} \cong \mathbf{D}_1 + \mathbf{D}_2 + \ldots + \mathbf{D}_Q \tag{3.1}$$

where each $\mathbf{D}_q$ $(q = 1, \ldots, Q)$ matrix satisfies the u.i. We use an overall *alternating least squares* (ALS) strategy to fit the mixture of tree structures. In particular, given current fixed estimates of all Q matrices except $\mathbf{D}_r$, we may define

$$\mathbf{\Delta}_r^* = \mathbf{\Delta} - \sum_{q \neq r}^{Q} \mathbf{D}_q \tag{3.2}$$

and use the mathematical programming procedures due to Carroll and Pruzansky described earlier, or one of the improved procedures due to De Soete [**16, 17**], to fit a least squares estimate, $\hat{\mathbf{D}}_r$, to $\mathbf{\Delta}_r^*$ (also see [**19**]). This combination of a mathematical programming approach to fitting individual ultrametric trees embedded within an alternating least squares "outer iterative" procedure is continued until convergence occurs, based on a criterion defined in terms of amount of change in the OLS fit measure.

3.3. Fitting of Path Length (or Additive) Trees. Farris, Kluge, and Eckart ([**21**]) and Hartigan [**23**] have shown that it is possible to convert a path length (or additive) tree into an ultrametric tree by a simple operation, given the distances from the root node to each of the nodes corresponding to objects. Letting d_{iR} represent the distance from the i-th object to the root node and d_{ij} be the path

length distance between i and j, it can be shown that:

$$\tilde{d}_{ij} = d_{ij} - d_{iR} - d_{jR} \qquad (\text{for } i \neq j) \tag{3.3}$$

satisfies the ultrametric inequality, although $\tilde{d}_{ij}$, will not, generally, satisfy the positivity condition for distances. However, both the ultrametric inequality *and* positivity can usually be satisfied by adding a sufficiently large constant K (see [2] for a precise statement of the conditions under which this is possible), thus defining d^*_{ij}, an ultrametric, as

$$d^*_{ij} = d_{ij} - d_{iR} - d_{jR} + K = d_{ij} - s_i - s_j \qquad (\text{for } i \neq j) \tag{3.4}$$

where $s_i = d_{iR} - K/2$. An equivalent statement is that

$$d_{ij} = d^*_{ij} + s_i + s_j \qquad (\text{for } i \neq j) \tag{3.5}$$

which states that the path length distance matrix $\mathbf{D}$ is decomposable into an ultrametric distance matrix plus an additive residual, which we shall simply call $\mathbf{S} \equiv (s_i + s_j)$, where, however, the diagonals of $\mathbf{S}$ are undefined, or zero if defined. When the decomposition can be defined so that all s_i are nonnegative, $\mathbf{S}$ is the distance matrix corresponding to a very special path length tree, usually called a "bush" by numerical taxonomists, or a "star" by graph theorists. (The latter terminology will be used henceforth.) A star tree is a path length tree with only one nonterminal node. The nonnegative constant s_i is, then, just the length of the branch connecting terminal node i to that single nonterminal node, while the distance between any two distinct terminal nodes, i and j, of the star, equals $s_i + s_j$. Thus we may summarize (3.5) verbally as follows:

A Path-Length Tree = An Ultrametric Tree + A Star.

We can also write this maxim as:

$$\mathbf{P} = \mathbf{U} + \mathbf{S}$$

where $\mathbf{P}$ is a path length distance matrix, $\mathbf{U}$ an ultrametric distance matrix, and $\mathbf{S}$ a distance matrix for a star tree. It should be noted that this decomposition is not unique. Given a fixed path length tree (PLT) there are many different ways of decomposing it into such a sum. In the case of multiple PLT's, since the sum of Q star trees is itself just a *single* star tree we have the extended result that:

A Sum of PLT's = A Sum of UT's + One Star.

Analogous to the matrix statement for single trees, this can be expressed as:

$$\sum_{q=1}^{Q} \mathbf{P}_q = \sum_{q=1}^{Q} \mathbf{U}_q + \mathbf{S} \tag{3.6}$$

It follows that we may fit mixtures of path length trees by simply adding to the ALS strategy defined earlier, an additional step in which the constants s_i, defining the single star tree component, are estimated by least squares procedures. Details of this and of the procedure implementing the estimation can be found in Carroll and Pruzansky [11].

Another approach that could be taken to fitting multiple path length trees would be to use an alternating least squares approach replacing the iterative fitting of ultrametric trees with the iterative fitting of path length trees, say by utilizing one of De Soete's [15, 18] procedures, or, perhaps by iterative fitting of a single path length tree and $Q - 1$ ultrametric trees, by use of a combination of mathematical

programming procedures for fitting these two different tree metrics (only *one* of which need be a path length metric).

4. A Heuristic Approach to Fitting Multiple Trees

While the mathematical programming/ALS approach described above will generally produce at least a *locally optimal* least squares solution, it is extremely time consuming *and* subject to a fairly serious local minimum problem. For this reason Pruzansky and Carroll [**28**] devised a heuristic approach for fitting multiple *ultrametric* trees that is much more efficient in computer time, and seemingly about as good from the point of view of avoiding merely locally (but non-globally) optimal solutions as the approach based on mathematical programming described above. While space constraints prohibit describing it thoroughly, the essential feature of this "heuristic" approach is replacement of the mathematical programming procedure for least squares fitting of a *single* ultrametric tree with a method that appears in practice to produce a very close approximation to the least squares ultrametric tree—namely "average linkage" (sometimes called UPGMA clustering). While average linkage does not, in general, find the exact best least squares tree, the continuous parameters ("height" values) it defines can easily be shown to be optimal from a least squares point of view, *conditional* on the particular tree. That is to say that, while the topology of the resulting tree may be suboptimal, the particular ultrametric fit based on that tree topology is (conditionally) optimal. Furthermore, it has been found that average linkage usually yields a tree with a topology that is very close to that of the optimal (least squares) tree. Thus our heuristic approach merely substitutes an average linkage fitting step for the much more expensive mathematical programming step within the context of the overall ALS approach. We might dub this AQLS, for Alternating Quasi-Least Squares, since each inner estimation step is only approximately (quasi-) least squares. An additional "twist" that seemed to improve the behavior of this heuristic algorithm in many ways was the introduction of an "easing" factor during the AQLS steps. In essence this "easing" factor amounts to subtracting from the data (dissimilarity) matrix not the total sum of distances from those trees already estimated, but a factor β_t times that sum ($0 < \beta_t \leq 1$) where t is an iteration index. Typically Pruzansky and Carroll began with β_t considerably less than 1 (usually somewhere between .65 and .95) and increased β_t gradually toward a value of 1 as iterations proceeded.

This "heuristic" approach can be extended to fitting single or multiple path length or additive trees by adding a phase in which the **S** matrix (for the "star" component) is estimated. Since there is an exact (analytical) least squares solution for **S**, this particular step is quite straightforward.

A final algorithmic note on this "heuristic" approach: even though it will not generally obtain even "locally" optimal least squares solutions ("locally" defined here in terms of a vaguely specified set of "local" combinatorial operations on trees) it may well provide a very good starting point for a more numerically sophisticated algorithm such as those described earlier involving use of mathematical programming techniques. Another heuristic approach to fitting trees or multiple trees is discussed by Hubert and Arabie [**24**].

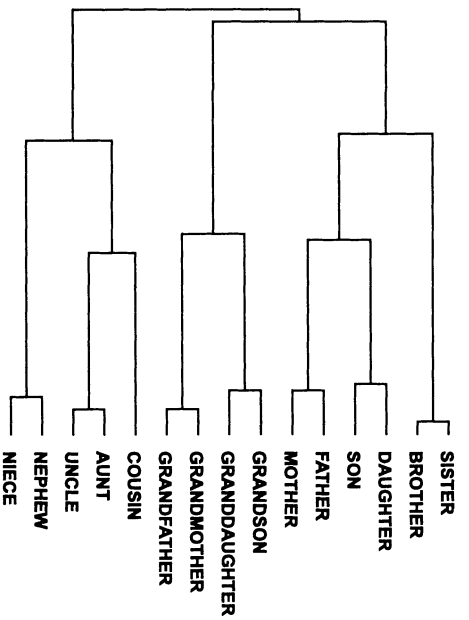


FIGURE 1. Multiple Tree Representation of the Kinship Data: First Tree.

5. Application of a Multiple Tree Model to Kinship Data

Illustrative applications of both the multiple trees and the hybrid approach are provided by some data from a study by Rosenberg and Kim [**30**], kindly provided by Seymour Rosenberg. These data were based on subjective sortings of kinship terms. Rosenberg and Kim applied standard hierarchical clustering techniques, but found that a dimension for gender failed to emerge although they knew it to be present from previous MDS (multidimensional scaling) analyses as well as simple inspection of the raw data. While a majority of subjects ignored gender in making their sortings (for example, always sorting "mother" and "father" together and "son" and "daughter" together) a minority of the subjects clearly used that dimension as the principal basis for sorting. The structure used by a majority of the subjects completely dominated and masked that used by the minority. MDS analysis, using the INDSCAL [**6**] approach for three-way, individual differences MDS, had also shown the gender dimension clearly to be present. It appeared that a two tree structure model would be appropriate, and indeed it did capture the essential features of the data quite nicely. (It might be noted that, for these data at least, essentially identical solutions were obtained using the mathematical programming based and "heuristic" approaches described earlier.) The first tree, shown in Fig. 1 corresponds very well to a standard anthropological model, the Romney–D'Andrade model [**29**], for kinship terms.

The major branch distinguishes between the "direct" relatives, lineals (who are in the same line of descent) and siblings, from the "collaterals," colineals (e.g., uncle, aunt, nephew, niece) and ablineals (cousins). Within each branch there is a further breakdown based on what might be called "absolute generation" (that is generational distance from "ego," or oneself). Within the "directs" we have a node

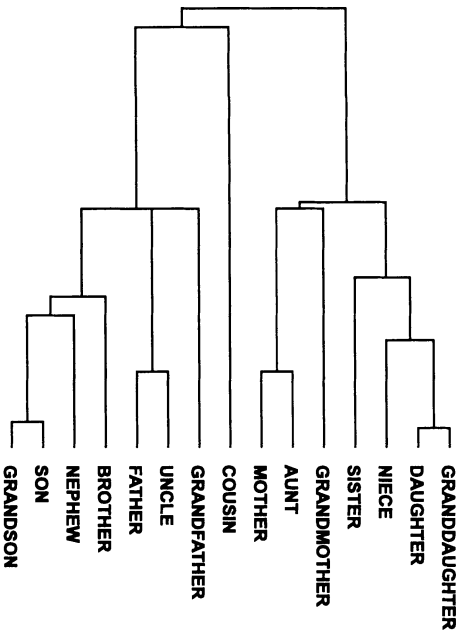


FIGURE 2. Multiple Tree Representation of the Kinship Data: Second Tree.

representing siblings, who are of the same generation as ego, a node representing children (daughter, son) or parents (father, mother), that is, lineals one generation removed from ego, and a node representing those lineals two generations removed from ego (grandchildren and grandparents). The collateral branch splits into those one generation above ego (aunt, uncle) those one generation below ego (nephew, niece) with the term at the same generational level (cousin) in some sense "between" those two nodes. Gender is nowhere in evidence as a basis for this tree; the analogous kinship terms of opposite sex are always together at the same node.

However, the second tree, seen in Fig. 2 has its main division based on gender, all the male terms at one principal node, the female terms at another, with cousin, the only genderless term, forming a third (singleton) branch. Some of the structure within each of the two main clusters seems to be related to generation.

6. Two Trees Solution for Similarities of Dyadic Relationships

Another multiple tree analysis, originally reported in Carroll [4] and also discussed in Carroll and Pruzansky [11], was of some data collected by Wish, Deutsch, and Kaplan [34] on various kinds of dyadic relationships. Wish et al. asked subjects to judge, on a nine point rating scale, the similarity among each pair of these dyadic relationships. (A typical judgment would involve rating the similarity of the relationship between, say, "mother-in-law and son-in-law" and that between "personal enemies.") Wish [33] carried out many analyses, among which an INDSCAL analysis ([6]) of the three-way array of data for all subjects, which resulted in a four dimensional solution. Wish et al. interpreted dimension 1 as "cooperation and harmony vs. competition and conflict," dimension 2 as "equality vs. inequality,"

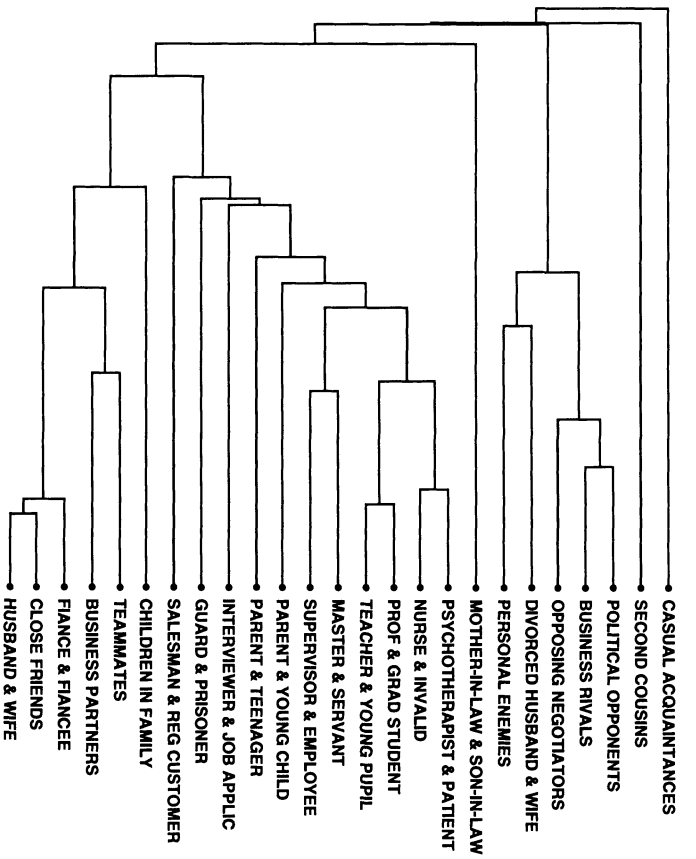


FIGURE 3. Multiple Tree Representation of the Dyadic Relationships Data: First Tree.

dimension 3 as "personal and informal vs. impersonal and formal" and dimension 4 as "intense vs. superficial." These interpretations were supported by evidence based on rating scale judgments.

The multiple tree approach was applied to these data resulting in a two ultrametric tree structure that seemed to conform remarkably well with the INDSCAL results. In fact, there seems to be a direct relation between each of the trees and one of the two planes of the four dimensional stimulus space from INDSCAL. For the first tree (shown in Fig. 3), the major branch appears to divide the cooperative or only mildly competitive relations in the subtree on the right from the more competitive ones on the left ("mother-in-law and son-in-law," "second cousins" and "casual acquaintances" not quite fitting into this scheme). There appears to be a mild anomaly in the location of "guard and prisoner," which winds up in the less competitive branch. However, the first INDSCAL dimension (see [**33, 11**]) confirms this finding—"guard and prisoner" is almost exactly at the same position on this dimension as is "parent and teenager." (In fact, "parent and teenager" and "guard and prisoner" are almost identically positioned on three out of four of the

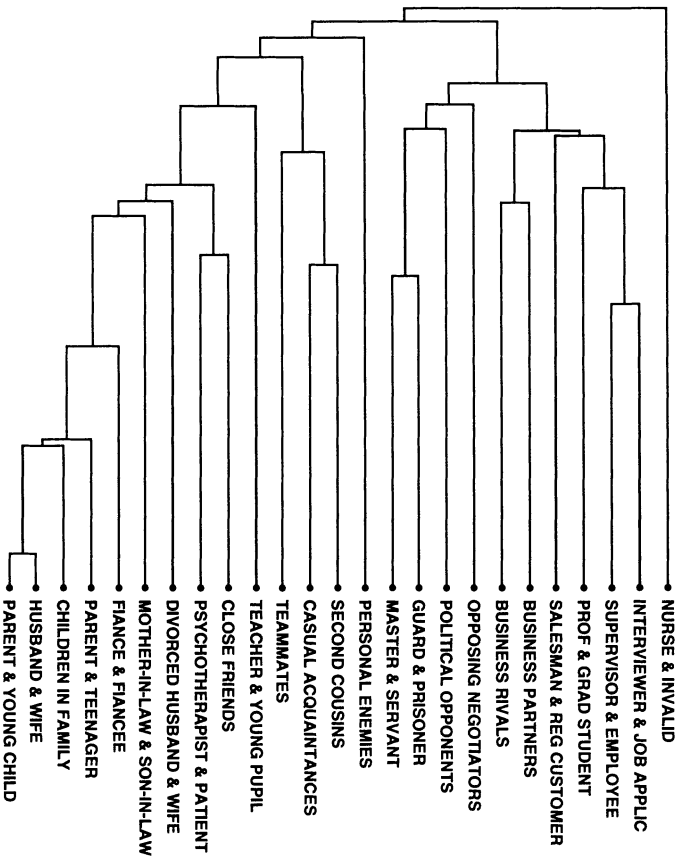


FIGURE 4. Multiple Tree Representation of the Dyadic Relationships Data: Second Tree.

INDSCAL dimensions—the exception being dimension three "personal and informal" versus "impersonal and formal," on which they differ radically. Perhaps this finding reveals something important about intergenerational relations!)

Within the cooperative, or less competitive branch, we see a further division between the more-or-less symmetric or equal relations (e.g., close friends, husband and wife, teammates) and the nonsymmetric or unequal ones (e.g., "professor and graduate student," "nurse and invalid" and "interviewer and job applicant"). There is no comparable division among the highly competitive relations, but this is not surprising. An inspection of the particular relations that are high in the competitive dimension, reveals they are all more or less equal or symmetric, as evidenced by the fact that they were grouped very close together on dimension 2 in the INDSCAL solution. Perhaps this is a matter of stimulus selection, or perhaps it is simply the nature of highly competitive relations that they are *ipso facto* symmetric or equal. To put it differently, unequals cannot compete, for the ascendancy of the dominant one of the pair makes competition impossible. (Maybe this also partly explains the seemingly anomalous position of "guard and prisoner.") Perhaps a better way to characterize this first tree is that it exhibits three main branches, corresponding

to "equal and cooperative," "equal and competitive," with the third branch being, simply, "unequal."

Some features of the tree structure are not obviously present in the dimensional representation. The extension of nodes specific to the relations, "teacher and young pupil" and "professor and graduate student" for example, or to the two relations "business rivals" and "political opponents" (just to pick two of a number that could be mentioned) suggest attributes unique to these relations that would be very difficult to capture in a low dimensional spatial model, but are easily incorporated into a tree structure or, especially, a *multiple* tree structure representation.

In an analogous way, the second tree, shown in Fig. 4, captures much of the structure in dimensions 3 and 4 of the INDSCAL solution. The left branch contains the personal and informal relations, while the right branch contains those that are more impersonal and formal (the two poles of the third INDSCAL dimension). Each of these then can be seen to split into "intense" versus "superficial," the extremes of INDSCAL dimension four. Of the personal and informal relations, "husband and wife" and "fiance and fiancee" are examples of intense relations, while "casual acquaintances" and "second cousins" are more superficial ones. Of the impersonal and formal relations, "guard and prisoner" and "political opponents" are examples of intense relations, while "interviewer and job applicant" and "salesman and regular customer" are more superficial ones. Again, there is a structure in the tree that is hard to represent spatially, and of course, there are ways in which the tree is inconsistent with the configuration in that plane. These two trees represent structure that conveys information not possible to incorporate into a low dimensional spatial representation, yet which complements that more continuous dimensional structure quite effectively.

7. Organizing Overlapping Clustering Representations into Trees or Multiple Trees

A number of methods have been proposed for deriving *overlapping* cluster structures from proximity data. Among the most widely used are procedures based on the ADCLUS (ADditive CLUStering) model, first proposed by Shepard and Arabie [**32**]. The most widely used method for *fitting* ADCLUS is the MAPCLUS procedure devised by Arabie and Carroll [**1**]. The ADCLUS model and MAPCLUS method were generalized to the three-way, individual differences case by Carroll and Arabie [**5**], in the form of the INDCLUS (INdividual Differences CLUStering) model and method. The INDCLUS method also allows fitting the two-way ADCLUS model as the two-way special case. An improved algorithm, called SINDCLUS, was later developed by Chaturvedi and Carroll [**12**]. All three of these methods are based on optimizing an OLS criterion of fit. Mirkin [**27**] has devised an approach, called "qualitative factor analysis," based on the same model, but using a sequential fitting procedure each step of which is *conditionally* OLS, but which does not, in general, optimize an overall OLS loss function.

In the INDCLUS model for similarity data, s_{ijk}, the similarity between objects j and k as judged by subject (or measured for source of data) i, is of the form:

$$s_{ijk} = \sum_{c=1}^{C} w_{ic} p_{jc} p_{kc} \tag{7.1}$$

where

$$p_{jc} = \begin{cases} 1 & \text{if object } j \text{ belongs to cluster } c, \\ 0 & \text{otherwise.} \end{cases}$$

while w_{ic} is a nonnegative weight indicating the importance or salience of class or cluster c for subject/source i. Thus the matrix $\mathbf{P} \equiv (p_{jc})$ is a matrix of indicator variables specifying the membership of the objects $j = 1, 2, \ldots, J$ in (possibly overlapping) classes or clusters $c = 1, 2, \ldots, C$. The matrix $\mathbf{W}_i \equiv \text{diag}(w_{ic})$ is a diagonal matrix of weights for the i-th subject/source, whose diagonal entries are the weights indicating the importance or salience weights for this subject or source. In matrix notation, the INDCLUS model can be written as:

$$\mathbf{S}_i \cong \mathbf{P}\mathbf{W}_i\mathbf{P}' \tag{7.2}$$

where $\mathbf{S}_i \equiv (s_{ijk})$ is the $J \times J$ matrix of similarities for subject/source i. The two-way case is simply written without the "i" subscript as

$$\mathbf{S} \cong \mathbf{P}\mathbf{W}\mathbf{P}' \tag{7.3}$$

or, in the case of equation (7.1), by simply dropping subscript i from s_{ijk} and w_{ic}. The overlapping cluster structure encoded by $\mathbf{P}$ will be denoted O.

Since the structure of the INDCLUS model is of the same form, for each subject or source, as the two-way ADCLUS model, but with separate individual source/subject weights, we will limit our discussion henceforth to that two-way model for an ordinary two-way (but one-mode, or set of objects) matrix of similarity data, $\mathbf{S}$.

We can convert s_{jk} to a dissimilarity δ_{jk} by simply subtracting it from a large positive constant K, so that:

$$\delta_{jk} = \begin{cases} K - s_{jk} & \text{for } j \neq k, \\ 0 & \text{otherwise.} \end{cases} \tag{7.4}$$

(We assume K is larger than $\max(s_{jk})$ for all $j \neq k$, so δ_{jk} is positive for all $j \neq k$.) Let us suppose, now, that the overlapping cluster structure encoded by $\mathbf{P}$, while overlapping, is nested, so as to be consistent with a hierarchical structure normally modeled by a tree. We denote such a nested hierarchical structure by H. Then it can be shown that δ_{jk} will be an ultrametric defined on the tree structure consistent with H. Each cluster in H corresponds to an internal node of this tree. (Note that all clusters in ADCLUS must contain *at least two* objects, since a singleton cluster will not affect the similarity as defined in (7.1)–(7.3) above, so that terminal nodes—corresponding to single objects—are not included in the cluster structure defined by ADCLUS or INDCLUS.) The height values associated with these clusters/internal nodes can be shown to be

$$h(c) \equiv K - \sum_{c^* \in \mathcal{S}(c)} w_{c^*}, \tag{7.5}$$

where $\mathcal{S}(c)$ is the set of clusters (including c itself) that are supersets of c (i.e., $\mathcal{S}(c)$ is the set of clusters, $\{c^*\}$, in the nested clustering H, such that c is a subset of each c^*). This can easily be seen by noting that any object, j, that is a member of c, is also a member of every set in $\mathcal{S}(c)$, so if both $j, k \in c$, it is also the case that

$j, k \in c^*$, for all $c^* \in \mathcal{S}(c)$; therefore:

$$s_{jk} \geq \sum_{c^* \in \mathcal{S}(c)} w_{c^*} \,. \tag{7.6}$$

The inequality in (7.6) is an *equality* if and only if c^* is the "lowest common ancestor" cluster for j and k, that is the *smallest* cluster in the hierarchically nested cluster structure H such that *both* $j, k \in c^*$.

Thus, s_{jk}, the *similarity* between j and k, can be defined as:

$$s_{jk} = h^*(c_{jk}), \tag{7.7}$$

where

$$h^*(c) = \sum_{c^* \in \mathcal{S}(c)} w_{c^*}$$

while c_{jk} is the "lowest common ancestor" of j and k (the smallest cluster/class containing both). We can call a similarity measure s satisfying these conditions an "anti-ultrametric" since it is defined by "anti-heights" defined on the hierarchical structure, which satisfy the *obverse* of the partial ordering with which the "heights" defining an ultrametric on the associated tree structure must be consistent. Thus δ_{jk} as defined in (7.4) will be given by:

$$\delta_{jk} = K - s_{jk} = h(c_{jk})$$

where $h(c) = K - h^*(c)$ is the "height" of cluster c, or equivalently, the height of the node in the hierarchical tree representing the nested cluster structure H, as stated earlier. It should be noted, too, that the partial order with which the "heights" of the ultrametric are required to be consistent (i.e., that $h(c_1) \geq h(c_2)$ if $c_1 \supseteq c_2$), will hold for this definition.

While an overlapping cluster structure O will not necessarily comprise a nested structure (i.e., one having the property that every pair of distinct clusters are either disjoint or one is a proper subset of the other), any finite overlapping cluster structure can always be decomposed into the union of hierarchically nested cluster structures $H_1, H_2, \ldots, H_Q$, each of which *is* a nested structure. (This is trivially true, since a single cluster comprises, by definition, a nested cluster structure—so at worst the structure can be trivially decomposed into at most C such nested structures. Clearly, this decomposition is of interest only if $Q \ll C$.) We define $M = \{H_1, H_2, \ldots, H_Q\}$ to be a *minimal* tree covering of O *iff* M covers O while no covering with $Q' < Q$ nested structures exists. Note that a minimal tree covering is not necessarily unique, however. To assure that all these nested structures correspond to complete trees, whose terminal nodes comprise the total set of J objects, we automatically include the universal set, containing all J objects, and all singleton clusters, corresponding to the objects themselves, in every such structure. The universal set is a superset of *every* cluster, of course, so it defines the root node of all Q hierarchical trees.

It also follows that, if O has an ADCLUS based similarity structure defined on it, then the corresponding dissimilarity defined by $\delta_{jk} = K - s_{jk}$ can be decomposed into a sum of ultrametrics, so that

$$\mathbf{\Delta} = \mathbf{U}_1 + \mathbf{U}_2 + \ldots + \mathbf{U}_Q$$

where $\mathbf{U}_q$ is an ultrametric defined on the hierarchical tree associated with H_q. If two or more of the nested structures contain this same cluster, the weight associated

with that cluster would have to be divided in some way between (or among) the anti-ultrametric similarities associated with these structures, and this would imply a corresponding indeterminacy in defining the associated ultrametrics. It should be noted that the ultrametric inequality does not necessarily imply positivity or even nonnegativity of the associated values. To guarantee this property we must add a "sufficiently large" constant to each $\mathbf{D}_q$ matrix—but this can be accomplished by simply defining the constant K used to transform the similarity measure into a dissimilarity also to be "sufficiently large." Therefore, it follows that, with K sufficiently large, the derived dissimilarity defined on O can be shown to be a metric, which can be decomposed into a sum of Q ultrametrics, satisfying positivity in addition to the ultrametric inequality.

It remains to determine how to define a minimal tree covering (or set of minimal tree coverings, if not unique) for an overlapping cluster structure O. Recalling that a cluster structure is hierarchically nested if and only if every pair of clusters is either disjoint or one contains the other, we define the *nesting graph*, N, to be a graph whose vertices correspond to the C clusters in the structure O, and which has an edge connecting two vertices a and b iff *either* a and b are disjoint *or* one of the two is a proper subset of the other. As before, we can assume that the universal set or cluster is always a vertex of N. It will, necessarily, be connected by an edge to every other vertex of N. Every object—viewed as a singleton cluster—can also be considered a vertex of N, with an edge connecting it to every other vertex. We need not, however, include these $J+1$ additional clusters explicitly in N, since they all are necessarily included in *every* clique of N, and therefore in every tree in a tree covering. It follows that any subset of clusters that define a *complete* subgraph of N can be associated with a hierarchically nested structure H.

A maximal nested structure H—that is one such that adding any other cluster from O would make it a *non*-nested structure—will correspond to a maximal complete subgraph, or clique, of O. Thus a clique finding algorithm, which finds all cliques (or maximal complete subgraphs) of O, will thereby find all possible maximal hierarchical structures (or maximal trees) embedded in O. Since any tree structure embedded in O must be a subtree of a maximal tree, finding all cliques of the nesting graph N, implicitly finds all possible trees embedded in O. It follows, then, that a minimal tree covering of O can be found by finding the minimal *clique* covering of N. Since the minimal clique covering of a graph is not necessarily unique, there must be some other criterion used for choosing a particular minimal tree representation, from the set of all minimal clique coverings of N. Details of this, and some criteria that can be used to choose the specific minimal tree covering of O to use, can be found in Carroll and Corter [**9**].

It should be noted that, if one adds a distance matrix corresponding to a star tree to a dissimilarity matrix derived via (7.1) from an ADCLUS similarity structure, this augmented matrix can, as shown in the first part of this paper, be decomposed into a sum of ultrametrics plus a star metric—which, as we have seen, is equivalent to a sum of path length or additive trees. Carroll and Corter [**8**] in some unpublished work, devised an "extended MAPCLUS" algorithm which fits a similarity structure of exactly the obverse of this form (i.e., a sum of an ADCLUS similarity structure and additive constants which can be viewed as the negative of those defining a "star" tree). Overlapping cluster structures derived from an algorithm of this kind applied to similarity data could then be decomposed into

TABLE 1. MAPCLUS Solution of the Kinship Data.

Cluster	Weight	Elements
(1)	.582	Brother, Sister
(2)	.554	Father, Mother
(3)	.551	Daughter, Son
(4)	.547	Granddaughter, Grandfather, Grandmother, Grandson
(5)	.468	Aunt, Cousin, Nephew, Niece, Uncle
(6)	.432	Nephew, Niece
(7)	.398	Aunt, Daughter, Granddaughter, Grandmother, Mother, Niece, Sister
(8)	.395	Brother, Father, Grandfather, Grandson, Nephew, Son, Uncle
(9)	.311	Brother, Daughter, Father, Mother, Sister, Son

a multiple path length/additive tree structure representation of the dissimilarities defined via (7.4).

It should finally be noted that *any* overlapping class or cluster structure—no matter how defined—can be decomposed into a (logical) sum of hierarchical trees, whether or not these are associated with ultrametric or other tree distances in any sense, via use of the procedure introduced by Carroll and Corter [**9**], based on applying a clique finding algorithm (such as [**3**]) to the associated nesting graph to find minimal tree coverings.

Finally, we consider an application of this approach, as reported by Carroll and Corter [**9**], to fitting a two-tree structure representation for the Rosenberg and Kim [**30**] kinship data. Carroll and Corter [**9**] first used MAPCLUS to fit a nine cluster solution which is shown in Table 1, and then used the Bron–Kerbosch [**3**] algorithm to find a minimal clique covering of this overlapping cluster structure. In this particular case, the clique finding algorithm found *exactly* two cliques—so there was no problem at all in deciding on the optimal two-tree representation. In fact, these two cliques corresponded to the two ultrametric tree structure solution shown in Fig. 5. Details of this analysis, and further details of this general approach, including the central role played by the "nesting graph" of an overlapping cluster structure O, can be found in Carroll and Corter [**9**].

The two trees determined in this way exhibit a very similar structure to that in the two-tree representation for these data derived directly via the mathematical programming approach described earlier, and shown in Figures 1 and 2, except that the structure in these two trees is much "simpler" than that in the earlier two-tree solution. The reason is that the total number of clusters in the two hierarchical clusterings associated with the trees in Fig. 5 is much smaller (only 9, as opposed to a total of 26, or $2(J-2)$—excluding the universal set and the singleton clusters) in the solution derived via the mathematical programming procedures—which can, of course, always improve the fit, at least slightly, by using the largest possible number of clusters consistent with this representation. Thus, the use of this approach, combining fitting an overlapping clustering structure via a procedure such as MAPCLUS, INDCLUS or SINDCLUS to proximity data (with a relatively small number of clusters) can be viewed as a way of fitting *constrained* multiple tree structure

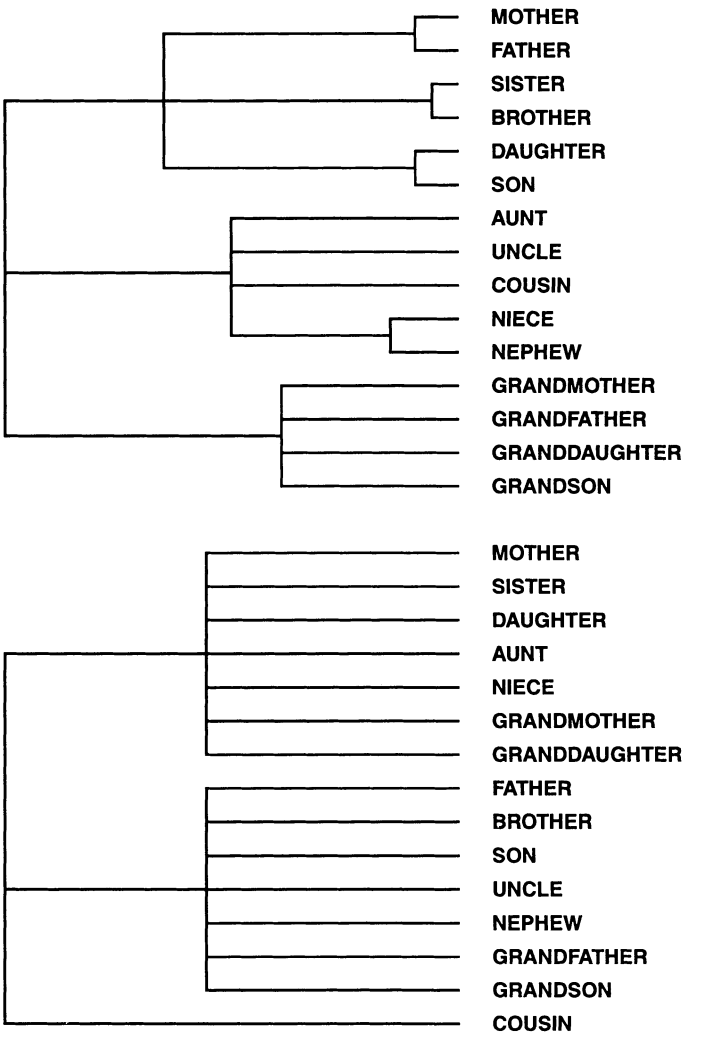


FIGURE 5. Two-Tree Representation of the Kinship Data.

representations—constrained to be "simple" or parsimonious in the sense that they induce a relatively small number of overlapping clusters associated with internal nodes. As described in Carroll and Corter [**9**], this general approach can also be used to fit other discrete representations to proximity data such as, for example, "extended similarity trees" [**13**].

References

[1] P. Arabie and J. D. Carroll, MAPCLUS: A mathematical programming approach to fitting the ADCLUS model. *Psychometrika*, 45: 211-235, 1980.

[2] J.-P. Barthélemy and A. Guénoche, *Trees and proximity representations* (G. Lawden, Trans.), New York, Wiley, 1991.

[3] C. Bron and J. Kerbosch, Finding all cliques of an undirected graph. *Comm. Assoc. Comput. Machn.*, 16: 575–577, 1971.

[4] J. D. Carroll, Spatial, non-spatial and hybrid models for scaling. *Psychometrika*, 41: 439–463, 1976.

[5] J. D. Carroll and P. Arabie, INDCLUS: An individual differences generalization of the ADCLUS model and the MAPCLUS algorithm. *Psychometrika*, 48: 157–169, 1983.

[6] J. D. Carroll and J. J. Chang, Analysis of individual differences in multidimensional scaling via an N-way generalization of "Eckart–Young" decomposition. *Psychometrika*, 35: 283–319, 1970.

[7] J. D. Carroll, J. D. and J. J. Chang, A method for fitting a class of hierarchical tree structure models to dissimilarities data and its application to some "body parts" data of Miller's. *Proceedings of the 81st Annual Convention of the American Psychological Association*, 8: 1097–1098, 1973.

[8] J. D. Carroll, J. D. and J. E. Corter, *CLUSTREES: A method for representing non-hierarchical cluster solutions as extended trees or multiple trees.* Unpublished manuscript, Bell Laboratories, Murray Hill, NJ, 1988.

[9] J. D. Carroll, J. D. and J. E. Corter, A graph-theoretic method for organizing overlapping clusters into trees, multiple trees, or extended trees. *J. Classification*, 12: 283–313, 1995.

[10] J. D. Carroll and S. Pruzansky, Fitting of hierarchical tree structure (HTS) models, mixtures of HTS models and hybrid models, via mathematical programming and alternating least squares. *Proceedings of the U.S.-Japan Seminar on Multidimensional Scaling*, 9–19, 1975.

[11] J. D. Carroll and S. Pruzansky, Discrete and hybrid scaling models. In *Similarity and choice*, E. D. Lantermann and H. Feger, Eds. Bern, Hans Huber, 1980, pp. 108–139.

[12] A. Chaturvedi and J. D. Carroll, An alternating combinatorial optimization approach to fitting the INDCLUS and generalized INDCLUS models. *J. Classification*, 11: 155–170, 1994.

[13] J. E. Corter and A. Tversky, Extended similarity trees. *Psychometrika*, 51: 429–451, 1986.

[14] J. P. Cunningham, Free trees and bidirectional trees as representations of psychological distance. *J. Math. Psych.*, 17: 165–188, 1978.

[15] G. De Soete, A least squares algorithm for fitting additive trees to proximity data. *Psychometrika*, 48: 621–26, 1983.

[16] G. De Soete, A least squares algorithm for fitting an ultrametric tree to a dissimilarity matrix. *Pattern Recognition Letters*, 2: 133–137, 1984.

[17] G. De Soete, Ultrametric tree representations of incomplete dissimilarity data. *J. Classification*, 1: 235–242, 1984.

[18] G. De Soete, Additive tree representations of incomplete dissimilarity data. *Quality & Quantity*, 18: 387–93, 1984.

[19] G. De Soete and J. D. Carroll, Tree and other network models for representing proximity data. In *Clustering and classification*, P. Arabie, L. J. Hubert, and G. De Soete, Eds. Singapore, World Scientific, 1996, pp. 157–197.

[20] J. S. Farris, Estimating phylogenetic trees from distance matrices. *American Naturalist,* 106: 645–668, 1972.

[21] J. S. Farris, A. G. Kluge, and M. J. Eckart, A numerical approach to phylogenetic systematics. *Systematic Zoology,* 19: 172–189.

[22] J. A. Hartigan, Representation of similarity matrices by trees. *J. Amer. Statist. Assoc.*, 62: 1140–1158, 1967.

[23] J. A. Hartigan, *Clustering algorithms*, New York, Wiley, 1975.

[24] L. J. Hubert and P. Arabie, Iterative projection strategies for the least squares fitting of tree structures to proximity data. *British J. Math. Statist. Psych.*, 48:281–317, 1995.

[25] C. J. Jardine, N. Jardine, and R. Sibson, The structure and construction of taxonomic hierarchies. *Math. Biosciences*, 1: 173–179.

[26] S. C. Johnson, Hierarchical clustering schemes. *Psychometrika*, 32: 241–254, 1967.

[27] B. Mirkin, Additive clustering and qualitative factor analysis methods for similarity matrices. *J. Classification*, 4: 7–31, 1987.

[28] S. Pruzansky and J. D. Carroll, *An analytical approach to fitting multiple hierarchical tree structures.* Paper presented at the 12th Annual Meeting of the Classification Society, Toronto, Canada, 1981.

[29] A. K. Romney and R. G. D'Andrade, Cognitive aspects of English kin terms. *American Anthropologist*, 66: 146–170, 1964.

[30] S. Rosenberg and M. P. Kim, The method of sorting as a data-gathering procedure in multivariate research. *Multivariate Behavioral Research*, 10: 489–502, 1975.

[31] S. Sattath and A. Tversky, Additive similarity trees. *Psychometrika*, 42: 319–345, 1977.
[32] R. N. Shepard, R. N. and P. Arabie, Additive clustering: Representation of similarities as combinations of discrete overlapping properties. *Psychological Review*, 86: 87–123, 1979.
[33] M. Wish, Comparisons among multidimensional structures of interpersonal relationships. *Multivariate Behavioral Research*, 11: 297–324, 1976.
[34] M. Wish, M. Deutsch, and S. J. Kaplan, Perceived dimensions of interpersonal relations. *Journal of Personality and Social Psychology*, 33: 409–420, 1976.

FACULTY OF MANAGEMENT, RUTGERS UNIVERSITY, MANAGEMENT EDUCATION CENTER, 125, 81 NEW STREET, NEWARK, NJ 07102-1895, USA
E-mail address: dcarroll@rci.rutgers.edu

DEPARTMENT OF DATA ANALYSIS, UNIVERSITY OF GHENT, HENRI DUNANTLAAN 1, B-9000 GHENT, BELGIUM
E-mail address: Geert.DeSoete@rug.ac.be

https://doi.org/10.1090/dimacs/037/23

DIMACS Series in Discrete Mathematics
and Theoretical Computer Science
Volume **37**, 1997

Linear Embedding of Binary Hierarchies and Its Applications

Boris Mirkin

ABSTRACT. The discrete binary hierarchy (DBH) is a concept underlying many important issues in analysis of complex systems: knowledge structures, test-and-search organization, evolutionary trees, taxonomy, data handling, etc. It appears that any DBH corresponds to an orthonormal basis of the Euclidean space related to the hierarchy leaves. The properties of these bases form a mathematical framework which can be applied to such problems as clustering and multiresolution image/signal processing. Clustering applications are based on a DBH-based analogue of the singular-value-decomposition of data matrices. A theoretical support for a method in divisive clustering is provided along with some decomposition-based interpretation aids. Data processing applications appear parallel to those involving the concepts of wavelets and quadtrees. However, DBH-based techniques seem to offer some potential improvements based on relaxing "continuity and homogeneity" restrictions of classical theories.

1. Introduction

The discrete binary hierarchy is a nested set of subsets ("clusters") of a finite N-element set such that any nonsingleton cluster is split in exactly two smaller clusters. It appears that any discrete binary hierarchy (in its ordered form) one-to-one corresponds to an orthonormal basis of the $N-1$-dimensional Euclidean space. The properties of these bases form a mathematical framework which is applied here to the problems of clustering and multiresolution image and signal processing.

In clustering, a divisive clustering strategy is substantiated as a method for the fitting of an approximation clustering model. The binary hierarchy provides for decompositions of the variance, covariance and the entries themselves via clusters, which gives additional interpretation aids to those usually employed in clustering. In image analysis, the binary hierarchy framework appears closely connected with some most exploited concepts, as wavelets and quadtrees, that correspond to "homogeneous and continouous" hierarchies. It should be expected that the binary hierarchies can lead to further advances in signal and image data processing by relaxing some restrictions of the classical approaches.

1991 *Mathematics Subject Classification.* 62H30, 90C27,05C50, 05C05.

The author thanks the Office of Naval Research for its support under grant number N00014-96-1-0208 to Rutgers University.

The remainder of the paper is arranged as follows. In Section 2, a linear embedding theory is outlined for discrete hierarchies: the concepts of hierarchy and ordered hierarchy are introduced in 2.1; three-value nest indicator functions and bases for binary and non-binary hierarchies are introduced in 2.2 and 2.3; the decompositions of the data via binary hierarchies are analyzed in 2.4; between-hierarchy transformations are considered in 2.5. Section 3 is devoted to hierarchical clustering: a binary-hierarchy based approximation clustering model is analyzed in 3.1 where a sequentially fitting approach is discussed as a method of divisive clustering. Two algorithms for splitting steps of the method are introduced in 3.2. An illustrative example is treated in 3.3, accompanied with many interpretation aids derived in Section 2. In Sections 4 and 5, potential applications for processing spatial data, both uni- and two-dimensional, are considered. In 4.1, the concepts of hierarchy layers and corresponding linear subspaces are introduced and used for data compression/decompression along the hierarchy. In 4.2, parallel concepts of wavelet-based multiresolution analysis theories are described. In 5.1, a concept of bihierarchy is introduced as a device for treating planar objects such as digitalized images. Its applications to clustering and fast compression/decompression on the plane are considered in 5.2 and 5.3, respectively. In Section 6, the main issues raised in the paper are outlined.

2. Hierarchies and Corresponding Orthonormal Bases

2.1. Hierarchies and Ordered Hierarchies. Hierarchies can be represented both in graph-theoretic and in set-theoretic terms. In this paper, only set-theoretic representation will be considered. Let I be a finite set consisting of N entities. A set of its subsets $S_W = \{S_w : S_w \subseteq I, w \in W\}$ called *clusters* is a *hierarchy* if it satisfies the following properties:

1. For any $i \in I$, $\{i\} \in S_W$;
2. $I \in S_W$;
3. The clusters S_w, $w \in W$, are nested, that is, $S_w \cap S_{w'} \in \{\emptyset, S_w\ S_{w'}\}$, for every $w, w' \in W$;

A hierarchy is a *binary hierarchy* if it satisfies the following additional condition:

4. For every non-singleton cluster S_w, $w \in W$, there exist two clusters $S_{w1}, S_{w2} \in S_W$ which are its proper subsets, such that $S_{w1} \cup S_{w2} = S_w$.

The definition implies that the clusters $S_{w1}, S_{w2} \in S_W$ in item 4 are defined in a unique way; sometimes they are referred to as *children* of cluster S_w which is considered their *parent*.

In graph-theoretic terms, a hierarchy is a leaf-labeled rooted tree; its nodes correspond to the clusters, and edges join the parents with their children. The root corresponds to I while the singletons to the leaves, each labeled with an entity $i \in I$. Every interior node, except for the root, is adjacent to at least three other nodes. In the binary hierarchies, every non-trivial cluster (that is, not a singleton or the root) is adjacent to exactly three nodes: its parent and children.

Obviously, the number of leaves equals N while the number of edges $N - 1$. For any binary hierarchy, $N - 1$ is also the number of its non-singleton clusters.

Three rooted trees in Fig.1 present two binary hierarchies because the clusters corresponding to the nodes of trees (a) and (c) are the same. A drawn (with no intersections) tree of a binary hierarchy is what can be called an *ordered hierarchy*: the children of every internal cluster are ordered with regard to each other so

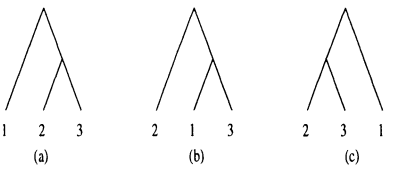


FIGURE 1. Three trees presenting two binary hierarchies on $I = \{1, 2, 3\}$.

that, say, the left child "precedes" the right one, according to this order. For any binary hierarchy, S_W, there are exactly $N-1$ non-singleton clusters and thus 2^{N-1} possible ordered versions, each corresponding to a drawn (with no edge crossings) representation of the hierarchy. Obviously, an ordered hierarchy corresponds with a linear ordering, P_W of I, defined so that iP_Wj iff in the minimum cluster $S_w \in S_W$ containing both i and j, the child containing i precedes (is drawn to the left of) the child containing j. All the ordered hierarchy clusters are intervals of this unique linear ordering, P_W, of I; that is, for any $S \in S_W$, $i, j \in S$ and iP_WkP_Wj implies $k \in S$. The tree in Fig.1(a) corresponds to the natural order, 123, and that in Fig.1(b) to the order 231. Conversely, given a linear ordering, P of I, such that all clusters of S_W are its intervals, implies that a corresponding tree can be drawn with no edge crossing. This can be put as follows.

STATEMENT 1. *A hierarchy, S_W, is ordered if and only if there exists a uniquely defined linear ordering, P_W, of I such that all the hierarchy clusters are its intervals and the hierarchy order is P_W trivially extended to clusters.*

2.2. Bases for Binary Hierarchies. Let S_W be an ordered binary hierarchy. For any nonsingleton cluster $S_w = S_{w1} \cup S_{w2}$ $(w, w1, w2 \in W)$ of S_W, its three-valued *nest indicator function* ϕ_w is defined as:

$$\phi_{iw} = \begin{cases} a_w & if \quad i \in S_{w1} \\ -b_w & if \quad i \in S_{w2} \\ 0 & if \quad i \notin S_w \end{cases} \tag{2.1}$$

where the reals a_w and b_w are well defined by the following conditions: (1) vector ϕ_w is centered; that is the sum of its components is zero; (2) vector ϕ_w has its norm, that is, the square root of the sum of its components squared, equal to 1, (3) S_{w1} precedes S_{w2} in the hierarchy order. To be more precise, let us denote by n_w, n_{w1}, n_{w2} the cardinalities of clusters S_w, S_{w1} and S_{w2}, respectively. Obviously, $n_{w1} + n_{w2} = n_w$. Then, (1) means that $n_{w1}a_w - n_{w2}b_w = 0$ while (2) gives $n_{w1}a_w^2 + n_{w2}b_w^2 = 1$. These two equations lead to the following values of a_w and b_w:

$$a_w = \sqrt{\frac{n_{w2}}{n_{w1}n_w}}, \; b_w = \sqrt{\frac{n_{w1}}{n_{w2}n_w}} \tag{2.2}$$

It turns out, the set of the vectors $\Phi_W = (\phi_w)$, $w \in W$, is an orthonormal basis of the (N-1)-dimensional space of all the $N-$dimensional centered vectors. Since the vectors ϕ_w are centered and normed by definition, it is sufficient to prove that these vectors are mutually orthogonal.

STATEMENT 2. *Every two vectors ϕ_w and $\phi_{w'}$ from the set $\Phi_W = (\phi_w)$, $w \in W$, defined for a binary hierarchy S_W by formula (2.1) are orthogonal; that is, their scalar product equals zero, $(\phi_w, \phi_{w'}) = 0$ $(w \neq w')$.*

Proof: Let us consider the scalar product $(\phi_w, \phi_{w'}) = \sum_{i \in I} \phi_{iw}\phi_{iw'}$. If $S_w \cap S_{w'} = \emptyset$ then $\phi_{iw}\phi_{iw'} = 0$ for any $i \in I$ since either $i \notin S_w$ or $i \notin S_{w'}$. Otherwise, one of the sets includes the other, say, $S_{w'} \subset S_w$, which implies that $S_{w'}$ is included in one of the children S_{w1}, S_{w2} of S_w, say, $S_{w'} \subseteq S_{w1}$. Then, $\phi_{iw} = a_w$ for any $i \in S_{w'}$, which implies that $\sum_{i \in I} \phi_{iw}\phi_{iw'} = a_w \sum_{i \in I} \phi_{iw'} = 0$, since vector $\phi_{w'}$ is centered. □

In matrix terms, the statement means that

$$\Phi^T \Phi = I_{N-1} \tag{2.3}$$

where I_n is the diagonal $n \times n$ identity matrix having all the diagonal entries equal to 1 and non-diagonal entries to 0.

It is not difficult also to prove that

$$\Phi \Phi^T = I_N - U/N \tag{2.4}$$

where U is the matrix having all its entries equal to 1 and, thus, each entry of U/N is equal to $1/N$. Equation (2.4) means that $\Phi\Phi^T$ is the orthogonal projector onto the subspace of all centered vectors.

The basis Φ_W can be considered as assigned to an unordered binary hierarchy. Since ordering subclusters, S_{w1} and S_{w2}, in this case is arbitrary, the matrix Φ corresponding to a binary hierarchy, S_W, is defined up to a right matrix factor, E, which is a diagonal matrix having its diagonal entries e_{ii} equal to 1 or -1 for any $i \in I$. The matrix ΦE corresponds to the same binary hierarchy as Φ, for any E defined above. This implies that every binary hierarchy can be ordered in 2^{N-1} ways.

2.3. Bases for Arbitrary Hierarchies. An orthonormal $(N-1)$-dimensional basis can be similarly defined for any hierarchy S_W. If $S_w \in S_W$ has $q \geq 3$ children $S_{wp} \in S_W$, $p = 1, ...q$, so that $S_w = S_{w1} \cup ... \cup S_{wq}$, a ternary nest indicator function can be defined for each of the children, S_{wp} (that is, for the edge between S_w and S_{wp}), as follows:

$$\phi_{iwp} = \begin{cases} a_{wp} & if \quad i \in S_{wp} \\ -b_{wp} & if \quad i \in S_w - S_{wp} \\ 0 & if \quad i \notin S_w \end{cases} \tag{2.5}$$

where the reals a_{wp} and b_{wp} satisfy the same conditions as above: $\sum_{i \in I} \phi_{iwp} = 0$, $\sum_{i \in I} \phi_{iwp}^2 = 1$, and S_{wp} is considered preceding $S_w - S_{wp}$. It is not difficult to prove that

$$a_{wp} = \sqrt{\frac{n_w - n_{wp}}{n_{wp} n_w}}, \; b_{wp} = \sqrt{\frac{n_{wp}}{(n_w - n_{wp}) n_w}} \tag{2.6}$$

Let us define a subset Φ of the nest indicator functions as follows. For every non-singleton $S_w \in S_W$, take in Φ all except one its nest indicator functions. It is not difficult to prove that Φ consists of exactly $N-1$ vectors.

STATEMENT 3. *The set Φ is a basis of the $(N-1)$-dimensional space of N-dimensional centered vectors. The nest indicator functions in Φ corresponding to non-siblings are mutually ortogonal.*

Proof: The same argument as in the proof of Statement 2 is applicable here except for the analysis of siblings which are absent from Φ, in the binary case. Let $S \in S_W$ consist of q children, $S_1, ..., S_q$, of which the functions $\phi_1, ..., \phi_{q-1}$ are in Φ and ϕ_q is not. Let $\sum_{p=1}^{q-1} \alpha_p \phi_p = 0$ for some reals, α_1,..., α_{q-1}. This sum, for an $i \in S_q$, equals $(-1/n) \sum_{p=1}^{q-1} \alpha_p \sqrt{\frac{n_p}{n-n_p}}$. For an $i \in S_1$, the sum differs by the first term only, which makes difference between these two values equal to $\alpha_1(\sqrt{\frac{n-n_1}{n_1}} - \sqrt{\frac{n_1}{n-n_1}})/\sqrt{n} = 0$. This implies that $\alpha_1 = 0$ if $n_1 \neq n/2$. The same is true for every coefficient α_p, $p = 1, ..., q-1$. Now let $n_1 = n/2$, which implies that equation $n_p = n/2$ is not true for any other $p < q$. Then $\alpha_2 = 0$. Considering difference of the sum values for $i \in S_1$ and $j \in S_2$ implies in this case that α_1 must be zero, too. Thus vectors $\phi_1, ..., \phi_{q-1}$ are linear independent. □

2.4. Decomposition of a Data Matrix via a Binary Hierarchy. Let us consider a $N \times n$ data matrix $Y = (y_{ik})$. Let us suppose all the columns $y_k = (y_{ik})$, $i \in I$, centered, that is, all the averages $\bar{y}_k = \sum_{i \in I} y_{ik}/N$ preliminarily subtracted from the components of corresponding column-vectors y_k, $k = 1, ..., K$.

Since every column-vector y_k, $k = 1, ..., K$ can be decomposed by the elements of basis Φ_W (for any ordered S_W), the following matrix equality holds:

$$Y = \Phi C \tag{2.7}$$

where $\Phi = (\phi_{iw})$ is the $N \times (N-1)$ matrix of values of the nest indicator functions in (2.1) and $C = (c_{wk})$ is a $(N-1) \times K$ matrix.

Since $\Phi^T \Phi$ is the identity matrix, multiplying the equality in (2.7) by Φ^T leads to

$$C = \Phi^T Y \tag{2.8}$$

which gives the value of every entry of matrix C expressed through the data as follows:

$$c_{wk} = \sum_{i \in I} \phi_{iw} y_{ik} = \sqrt{\frac{n_{w1} n_{w2}}{n_w}} (y_{w1k} - y_{w2k}) = \sqrt{\frac{n_{w1} n_w}{n_w - n_{w1}}} (y_{w1k} - y_{wk}) \tag{2.9}$$

where y_{wk}, y_{w1k} and y_{w2k} are the averages of the k-th variable in S_w, S_{w1} and S_{w2}, respectively.

It should be noted that this expression depends on the order of clusters in S_W: if Φ is changed for ΦE, then C is changed for EC. The latter expression in (2.9) is also valid for the basis corresponding to a non-binary hierarchy, as defined above.

Now consider a K-dimensional vector of the averages of the variables in a subset S_w, $w \in W$, and denote it by y_w. Then, the equality in (2.9) implies that the Euclidean norm $\sqrt{(c_w, c_w)}$ of the vector $c_w = (c_{wk})$ is equal to

$$\mu_w = \sqrt{\frac{n_{w1} n_{w2}}{n_w}} d(y_{w1}, y_{w2}) \tag{2.10}$$

where $d(x, y)$ is the Euclidean distance between vectors x, y. The value μ_w is positive if $x \neq y$, and zero if $x = y$. The norm is invariant to the between-cluster ordering and thus is well defined for nonordered binary hierarchies.

Defining M to be a diagonal $(N-1) \times (N-1)$ matrix with μ_w, $w \in W$, as its diagonal entries, and considering vectors c_w as being normed, the equation in (2.7) becomes an analogue of the singular-value decomposition (SVD) of the matrix Y

(see Golub and Van Loan (1989)) since, in this case, $Y = \Phi MC$ where Φ is matrix of an orthonormal vector set and M is a diagonal matrix with nonnegative diagonal entries. The weighted distances in (2.10) are analogues to the singular values; they will be referred to as the *cluster values* and the entries of C can be referred to as cluster loadings (by the analogy with the principal component anlysis loadings, Jolliffe, 1986).

For the sake of simplicity, the vectors c_w, $w \in W$, will not be considered normed, thus holding all the formulas above as they are.

On the other hand, the expression in (2.10) holds for any norm $||||$ as a function defined for the vectors $c_w, w \in W$, if the distance is accordingly defined as $d(y_{w1}, y_{w2}) = ||y_{w1} - y_{w2}||$. Moreover, the function $||||$ suffices to be any monotone function thus defining d as a dissimilarity measure which might fail to satisfy some metric properties (as the triangle inequality).

Another useful property of the equation (2.7) is that

$$Y^T Y = C^T C \tag{2.11}$$

which is proved by multiplying (2.7) with its transposed version since $\Phi^T \Phi$ is the identity matrix.

Equations (2.7) and (2.11) provide us with useful decompositions of the major data characteristics via the binary hierarchy clusters. This relates to: (a) variances of the variables, (b) between-variable covariations, and (c) the entries themselves. Since the columns of Y are centered, the elements (y_k, y_l) of the matrix $Y^T Y$ have the meaning of covariance (or even correlation) coefficients between the variables k and l (multiplied by N). This allows equation (2.11) to be rewritten using formula (2.9) as follows:

$$(y_k, y_l) = \sum_{w \in W} \frac{n_{w1} n_{w2}}{n_w} (y_{w1k} - y_{w2k})(y_{w1l} - y_{w2l}). \tag{2.12}$$

When $k = l$, we have the variance of the variable k decomposed by the clusters:

$$(y_k, y_k)/N = \sum_{w \in W} \frac{p_{w1} p_{w2}}{p_w} (y_{w1k} - y_{w2k})^2. \tag{2.13}$$

Summing up equations (2.13) and employing (2.10), we arrive at an equation

$$Tr(Y^T Y)/N = \sum_{i,k} y_{ik}^2/N = \sum_{w \in W} \frac{p_{w1} p_{w2}}{p_w} d^2(y_{w1}, y_{w2}) = \sum_{w \in W} \mu_w^2 \tag{2.14}$$

decomposing the squared data scatter (the total data variance) into the sum of cluster contributions which are the cluster values squared. The last decomposition (2.7) of the entries can be expressed using (2.9), as follows:

$$y_{ik} = \sum_{\{w1 : i \in w1\}} (y_{w1k} - y_{wk}) \tag{2.15}$$

where summing is applied to all filter of proper clusters, S_{w1}, containing i (S_w is the parent of S_{w1}).

According to (2.15), it is the between-center difference, $y_{w1k} - y_{wk}$, which characterizes the contribution of a cluster, S_{w1}, to the entries of all $i \in S_{w1}$.

All the four decompositions, (2.12) – (2.15), do not depend on an ordering of S_W. The decomposition (2.14) has been employed in clustering and (2.13), (2.15) in analysis of variance (ANOVA).

2.5. Transformation Matrices. Let Φ and Φ' be basis matrices corresponding to two ordered binary hierarchies on I. According to (2.7) and (2.8),

$$\Phi' = \Phi C$$

where

$$C = C(\Phi, \Phi') = \Phi^T \Phi'.$$

The matrix $C(\Phi, \Phi')$ can be referred to as the *transformation matrix* (transforming Φ into Φ'). Exploiting the latter equation in (2.9), its entries can be expressed as:

$$c_{ww'} = \sqrt{\frac{nn_1n_1'}{n'n_2n_2'}}\left(\frac{n_2'}{n_1'}\Delta(w1') - \Delta(w2')\right) \tag{2.16}$$

where $\Delta(w')$ is the difference between the probability of $S'_{w'}$ in S_{w1} and that in S_w, and n, n_1, n_2 refer to the cardinalities of the cluster S_w and its subclusters (in the Φ-hierarchy) while n', n_1', n_2' relate to those of the cluster $S'_{w'}$ and its subclusters (in the Φ'-hierarchy). More precisely,

$$\Delta(w') = p(S'_{w'}/S_{w1}) - p(S'_{w'}/S_w)$$

where the conditional probability, $p(S/T)$, $S \subseteq T \subseteq I$, is defined, as usual, as $|S \cap T|/|T|$.

For any three binary hierarchies (not necessarily distinct), equation (2.4) implies

$$C(\Phi, \Phi'') = C(\Phi, \Phi')C(\Phi', \Phi'') \tag{2.17}$$

which makes the set of all the ordered binary hierarchy bases $\{\Phi\}$ a finite group since the transformation matrices are normal (that is, they "rotate" the space having their determinants equal to unity) and thus nonsingular.

Let us call two transformation matrices, C and D, as *order-equivalent* if

$$c_{ww'} = \begin{cases} d_{ww'} & \text{if} \quad (w, w') \in S_1 \times S_2 \cup (I - S_1) \times (I - S_2) \\ -d_{ww'} & \text{if} \quad (w, w') \in (I - S_1) \times S_2 \cup S_1 \times (I - S_2) \end{cases}$$

for some $S_1, S_2 \subseteq I$. Order-equivalence is obviously an equivalence relation. Its equivalence classes correspond to transformations between nonordered binary hierarchies.

STATEMENT 4. *For any two binary hierarchies, the set of all transformation matrices between their ordered versions, is an equivalence class of the order-equivalence relation.*

Statement 4 implies that the group of transformation matrices between ordered hierarchies factored with regard to the order-equivalence is not a group. Specifically, the order-equivalence classes are not closed with regard to multiplication. Let us consider, for example, hierarchies presented in Figure 1. Depending on their orderings, we may have the following transformations between them:

$$A = \begin{pmatrix} -1/2 & \sqrt{3}/2 \\ \sqrt{3}/2 & 1/2 \end{pmatrix}, \quad B = \begin{pmatrix} -1/2 & -\sqrt{3}/2 \\ -\sqrt{3}/2 & 1/2 \end{pmatrix}$$

Matrix A corresponds to the orders (123) in (a) and (213) in (b) while the orders for B are (132) and (231), respectively. Obviously, $A^TA = A^2$ and $B^TB = B^2$ are equal to the identity matrix while

$$A^TB = AB = \begin{pmatrix} -1/2 & \sqrt{3}/2 \\ -\sqrt{3}/2 & 1/2 \end{pmatrix}$$

which is not the identity matrix. Neither does it belong to the group of transformation matrices: its determinant is $1/2$, and so it is not normal.

However, the absolute values of the entries in all order-equivalent matrices are equal. This implies that transformation matrices can be used, for instance, to analyze the differences between hierarchies. It should be noted that the value (2.16) can be considered as a rather non-standard way for evaluation of between-cluster similarity. To illustrate the transformation matrices as an "analitical" device for representing geometrical differences, let us consider three hierarchies presented in Figure 2.

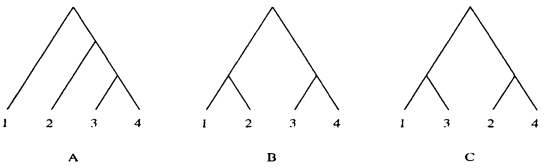


FIGURE 2. Three binary hierarchies over a four-element set.

The transformation matrices between them:

$$C(A,B) = \begin{pmatrix} \sqrt{1/3} & \sqrt{2/3} & 0 \\ \sqrt{2/3} & -\sqrt{1/3} & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad C(A,C) = \begin{pmatrix} \sqrt{1/3} & \sqrt{2/3} & 0 \\ -\sqrt{1/6} & \sqrt{1/3}/2 & \sqrt{3}/2 \\ \sqrt{1/2} & -1/2 & 1/2 \end{pmatrix},$$

and

$$C(B,C) = \begin{pmatrix} 0 & \sqrt{1/2} & \sqrt{1/2} \\ \sqrt{1/2} & 1/2 & -1/2 \\ \sqrt{1/2} & -1/2 & 1/2 \end{pmatrix}.$$

Obviously, the first matrix is somewhat simpler: hierarchies A and B in Figure 2 have a common cluster, $\{3,4\}$.

It seems quite natural to evaluate the overall between-hierarchy difference by a norm of the transformation matrix. However, the Euclidean norm, $Tr(C^TC) = \sum_{w,w'} c^2_{ww'}$ cannot do the job, because it is constantly equal to $N-1$, for any two hierarchies, as follows from (2.11) and the definition of Φ. Moreover, it can be easily proven that the matrices of squared entries of transformation matrices, $(c^2_{ww'})$, are doubly stochastic: the sum of elements in every column or row of such a matrix equal to 1.

The other norms are still available. In our example, the sums of the entries' absolute values (norm L_1) are equal to 3.79, 4.66, and 4.83, respectively, which seems in line with our intution in pair-wise comparisons of the trees in Figure 2.

3. Application to Hierarchical Clustering

3.1. Approximation Clustering Model. Traditionally, clustering is considered a discipline devoted to finding "cohesive" groups of points in a geometric space. Such a direct goal can be underlied with theoretical considerations of which we are interested in approximating. In this approach, the observed data is considered as a noisy information on the underlying discrete cluster structure. In such a setting the clustering problem is a problem of approximation of the noisy data with an adequate clustering structure (Hartigan (1972), Shepard and Arabie (1979), De Sarbo (1982), Mirkin (1990, 1994), Chaturvedi and Carroll (1994), Mirkin, Arabie, and Hubert (1995), etc.).

Let us refer to a set of subsets, $S_{W'}$, as a *hierarchical cluster structure* if it satisfies requirement (3) in the definition of binary hierarchy so that clusters in $S_{W'}$ are nested, though (some) singletons or/and the root, I, may be not in $S_{W'}$. A graph corresponding to a hierarchical cluster structure is a forest being only a part of a binary hierarchy tree; the leaves of the forest correspond to inclusion-minimal clusters in $S_{W'}$. The matrix of the nest indicator functions of non-leaf clusters in $S_{W'}$ will be denoted by Φ'. Obviously, any hierarchical cluster structure can be completed into a binary hierarchy by further partitioning its minimal non-singleton clusters and pair-wise merging its maximal clusters.

Representing hierarchical cluster structures with nest indicator functions, we arrive at the following approximation clustering model:

$$Y = \Phi' C' + E \tag{3.1}$$

where Φ' stands for a current hierarchical cluster structure, C' is an unknown matrix of cluster loadings and E is the matrix of residuals to be minimized with regard to arbitrary C' and admissible Φ'.

The least-squares criterion,

$$D(\Phi', C') = Tr[(Y - \Phi' C')^T (Y - \Phi' C')] = \sum_{i=1}^{N} \sum_{k=1}^{M} (y_{ik} - \sum_{w \in W'} \phi_{iw} c_{wk})^2 \tag{3.2}$$

will be the only scalar measure of the residuals considered in this paper.

STATEMENT 5. *Given a hierarchical cluster structure, Φ, the least-squares estimate for C' is determined by formula $C'(\Phi') = \Phi'^T Y$ which is analogous to (2.8). The minimum value of $D(\Phi', C')$ equals*

$$D(\Phi', C'(\Phi')) = Tr(Y^T Y) - \sum_{w \in W'} \mu_w^2 \tag{3.3}$$

where cluster values μ_w for non-leaf clusters are defined by formula (2.9) and are zeros for the leaf (minimal) clusters.

Proof: The equation $C' = \Phi'^T Y$ is derived as a necessary condition for minimality of (3.2). Putting this into (3.2), the equality $D(\Phi', C'(\Phi')) = Tr(Y^T Y - C'^T C')$ follows. Equations $Tr(C'^T C') = \sum_{w \in W'} \sum_k c_{wk}^2$ and (2.14) prove the statement. □

In fact, formula (2.14) gives decomposition of the squared data scatter, $Tr(Y^T Y)$, in two parts: explained, $\sum_{w \in W'} \mu_w^2$ and non-explained, $D(\Phi', C'(\Phi'))$,

by the cluster structure Φ'. An important feature of the formula $C'(\Phi') = \Phi'^T Y$ is that it holds only when the least-squares approximation is considered while the generic equality (2.8) holds always.

Let us define a set, A, of admissible hierarchical cluster structures $S_{W'}$ by the following two conditions: (a) $I \in S_{W'}$, so that the structure is a tree, not forest, and (b) the number of clusters, $|W'|$ is fixed. When $|W'| = 2N - 1$, the set A consists of all binary hierarchies, so we consider $|W'| < 2N - 1$.

According to equation (3.3), any least-squares fit to the model (3.1) must maximize the criterion

$$\sum_{w \in W'} \mu_w^2 = \sum_{w \in W'} \frac{p_{w1} p_{w2}}{p_w} d^2(y_{w1}, y_{w2})$$

so that the problem is to find $|W'|$ consecutive divisions of I maximizing the sum of the weighted between-center distances $d^2(y_{w1}, y_{w2})$.

The author has no nontrivial suggestions on globally resolving the problem. A major issue here is that it is unknown whether the optimal structures satisfy the so-called minimal distance rule or not. The minimal distance rule requires that the distance from any point in any cluster to the cluster's center is smaller than to the center of any other cluster. This rule drastically reduces the number of potential cluster structures to check.

Thus we suggest a greedy-wise procedure of sequential extraction of clusters from the data according to the least-squares criterion. This procedure is analogous to the standard one-by-one extraction procedure of the principal component analysis and described, in a general form called the SEFIT algorithm, in Mirkin, 1990.

At each iteration of SEFIT, w, the input information includes the subtree S'_W available (initially, $S'_W = \{I\}$) and a data matrix, Y, updated. There are two steps at the iteration, according to the general procedure: (1) updating S'_W by splitting a leaf-cluster to maximize the cluster contribution, μ_w^2, added; (2) updating Y by subtracting the item found, $y_{ik} \leftarrow y_{ik} - c_{wk}\phi_{iw}$. The computation ends when w reaches a pre-specified number of clusters.

Curiously, there is no need in step (2) of updating the data matrix since the value maximized at step (1),

$$\mu_w^2 = \frac{n_{w1} n_{w2}}{n_w} d^2(y_{w1}, y_{w2}) \tag{3.4}$$

is invariant with respect to subtracting cluster values from larger clusters, because $d(x, y) = d(x - a, y - a)$ for any real a.

Thus, SEFIT in this context reduces to what has been known in the clustering discipline as the divisive clustering with splitting criterion (3.4). This criterion is well known in clustering. Ward (1963) is credited for introducing it in the agglomerative clustering context; Edwards and Cavalli-Sforza (1965) have considered the same criterion for divisive clustering. Gower (1967) provided an example demonstrating a peculiarity of the criterion reflecting the fact that factor $n_{w1} n_{w2}/n_w$ in (3.4) favors equal distribution of the entities among the clusters and, thus, the criterion may fail to immediately separate some outliers. Though for a long time treated as a shortcoming, the peculiarity does not appear to actually be so. Moreover, in many clustering studies, tendency of the cluster cardinalities to the same number

has been suggested as a good criterion of clustering (see, for example, Braverman and Muchnik, 1983).

Let us consider how the criterion (3.4) can be applied in the case when all the variables are binary descriptors of qualitative categories represented by zero-one columns (one for Yes, zero for No).

Let us compute the within-cluster average of a zero-one variable k. Do not forget, that the variable has been centered initially, which means that the entries $1 - p_k$ and $-p_k$ stand for 1 and 0, respectively, where p_k denotes the relative frequency of ones in column k.

Thus, the average is $y_{wk} = (1 - p_k)p_{wk}/p_w - p_k(1 - p_{wk}/p_w)$ where p_{wk} is the frequency of simultaneously observing descriptor k and cluster S_w and p_w is the frequency of S_w. This leads to:

$$y_{wk} = p_{wk}/p_w - p_k. \tag{3.5}$$

which implies

$$c^2_{wk} = \frac{n_{w1}n_{w2}}{n_w}\left(\frac{p_{w1k}}{p_{w1}} - \frac{p_{w2k}}{p_{w2}}\right)^2 \tag{3.6}$$

This looks quite natural: the first factor "takes care" to get the split closer to halving (which corresponds to the information concepts of the search theory) while the second follows the difference between the frequencies of ones in the subclusters. It should be noted that this measure closely resembles the so-called "twoing rule" measure used in CART techniques for conceptual clustering; see Breiman et al. (1984), p. 38, 107, 127-129.

The criterion (3.4) in this case is just the weighted distance between within-cluster probability profiles:

$$\mu^2_w = \frac{n_{w1}n_{w2}}{n_w} d^2(p(w1), p(w2)) \tag{3.7}$$

where $p(w)$ is the vector of (conditional) probabilities of categories k in cluster S_w.

This shows that the least-squares criterion can be employed for clustering not only when all the variables are quantitative, but also when there are nominal variables present. Curiously, the formulas above fit into the problem of (multiple) alignment of biological sequences in the so-called continuous sequence space (Vingron and Sibbald, 1993). Basically, this space can be considered as a nominal data table where variables correspond to sequence positions and the categories are letters of a biomolecular alphabet.

3.2. Splitting Algorithms. Let us consider the step of splitting of a cluster S_w, in the divisive strategy, in more detail. Depending on the formula for c_{wk} in (2.9), the value of the maximized criterion μ^2 can be expressed by formula (3.4) or

$$\mu^2_w = \frac{n_w n_{w1}}{n_{w2}} d^2(y_{w1}, y_w) \tag{3.8}$$

Computationally, criterion (3.4)/(3.8) leads to a difficult, though not NP-complete splitting task. Indeed, as is well known, the optimal splits must satisfy the mimimal distance rule, which means that the convex halls of the subclusters are linearly separated. The number of splits generated by hyperplanes is known to be less than N^K (Bock, 1974) where K is the dimensionality of the variable space, which shows the complexity of the problem. Still no further reduction of complexity of the problem has been achieved.

We describe now two local search algorithms, for each of the two formulas for μ_w^2.

Formula (3.4) implies an algorithm which is just a version of the moving-center (K-Means) technique.

Splitting by Maximizing (3.4)
Initially, the most distant points y_1 and y_2 in S_w are determined to be used as the initial centers of the clusters.
Then, sequentially, the usual next two steps are performed iteratively: (a) assigning the entities to the clusters (the nearest center wins) and (b) recomputing the centers (as the centers of gravity of the clusters obtained in (a)). The computation ends when step (a) leaves the clusters unchanged.

Evidently, this version of the K-Means technique is nothing but the alternating minimization of the square-error clustering criterion (Jain and Dubes, 1988) by two groups of the variables, those related to membership of the entities to the clusters (a) and to the cluster centers (b). Simultaneously, it is an alternating maximization algorithm for the criterion (3.4).

The second algorithm, based on formula (3.8), is a seriation algorithm.

Splitting by Maximizing (3.8)
Initially, a point y_1 is found maximizing its distance to y_w, the center of S_w, to set $S_{w1} = \{y_1\}$. On a general step, a current S_{w1} along with its center y_{w1} is considered and an entity-point y_j, closest to y_{w1} by Euclidean distance, is sought. It is added to S_{w1} if the quotient $q = d^2(y_{w1}, y_w)/d^2$, where d is the distance between y_w and the center of $S_w \cup \{y_j\}$, satisfies the inequality

$$q < \frac{n_1 n_2 + n_2}{n_1 n_2 - n_1},$$

and the process ends if not.

The inequality involved is equivalent to the fact that value of μ_w^2 (3.8) increases when y_j is added to S_w. Basically, there is a trade-off between an increase of the coefficient n_{w1}/n_{w2} and correponding decrease of the distance $d^2(y_{w1}, y_w)$. The distance may only decrease in the adding process.

Though the analogy between the one-by-one strategy of principal component analysis and the square-error divisive clustering seems rather deep, any binary hierarchy defines a different SVD-like basis while there is only one genuine SVD basis consisting of the singular vectors employed in the principal component analysis.

The algorithms described can be extended to any dissimilarity function d and, thus, amount to a family of divisive clustering algorithms which overlap but not coincide with that of Lance and Williams (1967).

A computational strategy for divisive clustering, based on the theory above, can be set as follows:

1. Standardize the entity-to-variable data by shifting the origin into the point of the variable averages and norming the variables by a chosen norm.

2. Choose a dissimilarity function (it may be different from the distance driven by the norm chosen for standardizing).
3. Choose a clustering strategy (only the divisive one has been discussed above) and create a cluster hierarchy S_W with the strategy.
4. Draw a tree hierarchy representation reflecting the cluster values μ_w by the heights of the corresponding division nodes.
5. Interpret the hierarchy designed using:
   1) the drawn pattern of clustering;
   2) contributions of the clusters and cluster–variable pairs to the square scatter of the data as reflected in values of $\sum_{k=1}^{n} c_{wk}^2$ and c_{wk}^2 (2.9), respectively ($w \in W, k = 1, ..., K$);
   3) the cluster variable-to-variable covariance/correlations, $N_{w1}N_{w2}(y_{w1k} - y_{w2k})(y_{w1l} - y_{w2l})/N_w$, as items in the additive decomposition of the overall covariance (2.12);
   4) decompositions (2.15) of the entries y_{ik} by clusters.

3.3. An Illustrative Example. Let us consider data on sorting of terms related to the human body collected by G.A. Miller (1968) and reported in Rosenberg (1982). The natural hierarchy of the body parts should be reflected in the underlying cluster structure. The four variables represent dissimilarities of 16 body terms with "Head", "Arm", "Chest", and "Leg", respectively, as presented in Table 1. The hierarchical classification found with the divisive clustering algorithm at each

TABLE 1. An extract from Miller's sorting data (1968): number of subjects (out of 50) who did not put any given row-terms into the same category with 4 column-terms presented

i	Term	Head	Arm	Chest	Leg
1	Body	45	50	37	50
2	Cheek	19	50	49	50
3	Ear	18	49	50	49
4	Elbow	49	8	50	47
5	Face	14	48	47	48
6	Hand	48	14	50	46
7	Knee	49	47	50	8
8	Lip	18	49	50	49
9	Lung	48	49	17	49
10	Mouth	19	49	50	49
11	Neck	31	45	38	45
12	Palm	50	16	49	48
13	Thigh	47	45	48	5
14	Toe	49	47	50	13
15	Trunk	42	46	19	45
16	Waist	44	45	26	46

step maximizing the contribution to the total variance is presented in Figure 3 as indexed with the corresponding cluster values (reflected in the heights of the vertical edges). The squared cluster values μ_t^2, which are equal to contributions of the cluster divisions to the total variance, are presented (per cent) for contributing the most.

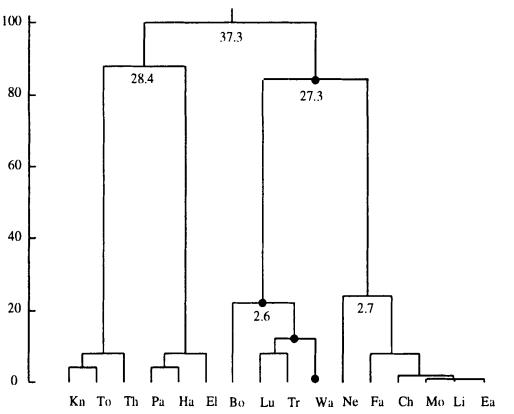


FIGURE 3. Binary hierarchy found for the data from Table 1 with the first splitting method for the least-squares criterion; the numbers show contributions of the major splits to the data variance.

The classification also leads to a decomposition of all the variances and correlations between the original variables. The general pattern of correlation is pair-wise negative as is seen in the correlation matrix:

1	1.00			
2	−0.48	1.00		
3	−0.24	−0.27	1.00	
4	−0.45	−0.15	−0.24	1.00
	1	2	3	4

Its decomposition by the first three separations, due to formula (2.12), is presented by the following respective matrix terms:

1	0.44			
2	−0.43	0.42		
3	0.32	−0.31	0.23	
4	−0.42	0.41	−0.30	0.40
	1	2	3	4

(first division),

1	0.00			
2	−0.01	0.56		
3	0.00	−0.01	0.00	
4	0.01	−0.57	0.01	0.58
	1	2	3	4

(second division),

1	0.43			
2	−0.02	0.00		
3	−0.54	0.02	0.60	
4	−0.01	0.00	0.02	0.00
	1	2	3	4

(third division).

These three items take into account most part of the variance and correlation. It can be seen, that all the variables are important for the first separation, although the third variable is somewhat less important (with its only 23% of the variance accounted) while the contribution of the first variable is some higher (44% of the variance). The second separation is due to the variables 2 and 4 while the third separation is made by the variables 1 and 3 (since the other variables in either case do not contribute to the variance at all).

Decomposition of the correlation coefficients confirms and details this conclusion. In particular, the negative correlations between the variables 1 and 3, as well as between 2 and 4, become positive at the first separation and sharper at the third and second separations, respectively. All the other correlations disappear in the clusters. The variance of variable 3 is not exhausted by the three first separations: this variable contributes to the separation of the smaller Head cluster.

The last interpretation aid concerns decomposition of all the standardized data entries y_{ik} by the clusters due to equation (2.15). Let us demonstrate the decomposition for the 16-th entity, Waist, belonging to the four clusters nested shown by the bold nodes in Figure 3:

1	0.52 =	−0.52+	1.09−	0.01−	0.05
2	0.28 =	0.50−	0.04−	0.06−	0.12
3	−1.46 =	−0.37−	1.20−	0.36+	0.47
4	0.36 =	0.49−	0.03−	0.05−	0.04

Every single column of the decomposition relates to its cluster reflecting the features of the cluster: the smaller values of the variables 1 and 3 in the first cluster correspond to its Head–Chest nature while the next cluster shows a split between these variables: enlarged 1 and reduced 3 correspond to the Chest membership of the entity. The last column represents individual traits of the entity.

Another tree (Figure 4) is generated with the divisive strategy when the criterion is changed for the so-called Chebyshev (uniform) metric and the second splitting algorithm has been applied. The data had been standardized as follows: the origin was shifted into the point of the average values of the variables, norming of the variables was performed by Chebyshev norm (the maximal absolute deviation from the average became one after norming was completed).

Contribution of the first split to the total variance in Figure 4 (44.9 %) is much higher than that in Figure 3 (37.3%)). This seems strange. How it could occur that the maximized contribution (in Figure 3) turned out less than the contribution achieved when another (Chebyshev) criterion was optimized (Figure 4)? To answer the question, let us consider decomposition of the variances of the variables by the

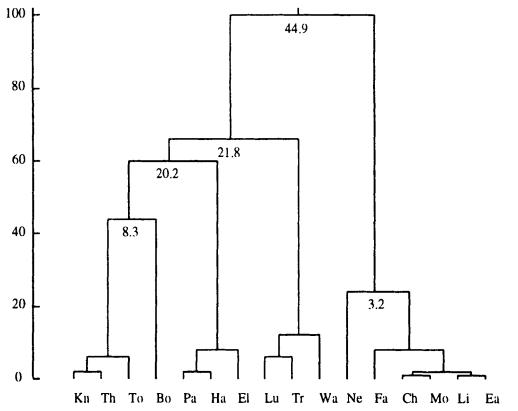


FIGURE 4. Binary hierarchy found for the data from Table 1 with the Chebyshev norm; the numbers show relative contributions of the major splits to the data variance.

clusters:

1	0.36 =	0.33+	0.00+	0.00 + ...
2	0.18 =	0.03+	0.02+	0.12 + ...
3	0.20 =	0.02+	0.15+	0.00 + ...
4	0.19 =	0.03+	0.03+	0.07 + ...

Again, only three major splits are represented in the decomposition. The variances (and, thus, the contributions to the square data scatter) of the variables now are different from the very beginning, which seems to determine the order they are involved in the division process: the most contributing variable 1 turns out to be the principal base of the first division; variable 3 having the second-large variance contributes mostly to the second division; the less contributing variables 3 and 4 are serving at the following divisions. Such a sequential involvement of the variables may generate a more complete account of the variance in splitting, which is reflected in the higher level of the variance extracted in the upper splits in Figure 4 as compared to those in Figure 3. This conclusion is supported by the results of the Euclidean-norm-based divisive clustering applied to the data standardized with Chebyshev norm (Figure 5). The variance contributions in the upper splits there are even greater (since the criterion is proper, in this instance); evidently, it is the left four-element cluster in Figure 4 disappearance which makes that increasing of the variances in Figure 5 possible. The contents of the clusters in the latter figure also seem quite satisfactory.

It looks that a general regularity is manifested in the example: Chebyshev norming generates a difference in the variances of the variables influencing the order of their involvement in splits (fusions) and thus increasing the contributions of the higher splits. This principle might cause the empirically observed facts that norming by range (which is quite similar to Chebyshev norming) made after centering by the average allows a best fit into Monte-Carlo generated cluster structures (Milligan and Cooper, 1988).

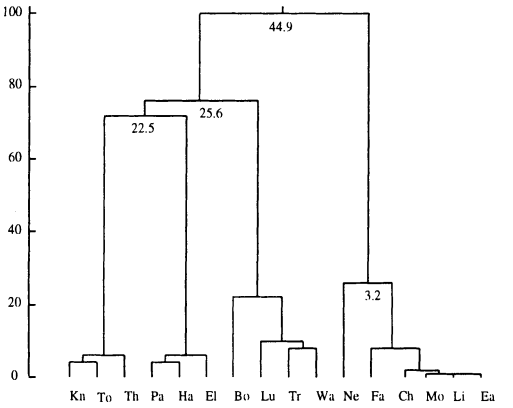


FIGURE 5. Binary hierarchy found with the least-squares criterion for the data from Table 1 normed by the Chebyshev norm

4. Application to Analysis of Spatial Data

4.1. Layered Hierarchies. The concept of ordered hierarchy fits into the so-called spatial data structures: digitalized intervals, rectangles or hyper-rectangles consisting of one-, two- or three- dimensional pixels ordered according to the coordinate axes (Samet, 1990). Let us initially consider I a unidimensional pixel set. An ordered binary hierarchy will be referred to as a spatial binary hierarchy if its order coinsides with the spatial ordering of I so that all clusters are unidimensional intervals as in the hierarchies A and B presented in Figure 6.

Any binary hierarchy can be equivalently represented by its decomposition into what will be called here layered hierarchy. A set of nested partitions of I, $\mathcal{L}=\{L^0, L^1, ..., L^n\}$, will be referred to as a *layered hierarchy* if (a) $L^0 = \{I\}$, (b) $L^n = \{\{i\} : i \in I\}$, (c) $L^m \subset L^{m-1}$, and (d) there exists a binary hierarchy, S_W, such that if S is a nonsingleton class of partition L^{m-1} then $S \in S_W$ and the children of S in S_W are classes of L^m $(m = 1, ..., n)$. Obviously, all classes in $\mathcal{L}$ are clusters of S_W and, moreover, there is an obvious one-to-one correspondence between the hierarchy S_W and layered hierarchy $\mathcal{L}$. The partitions $L^m \in \mathcal{L}$ will be called layers of the hierarchy.

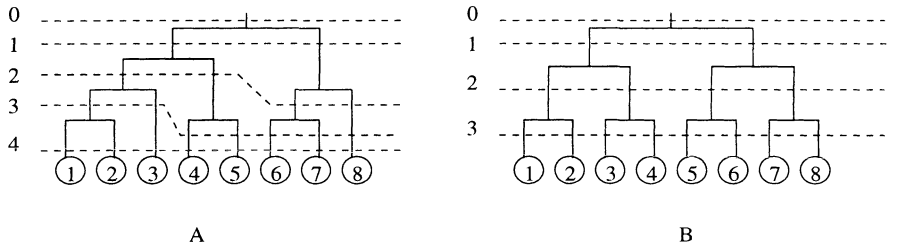


FIGURE 6. Two layered spatial binary hierarchies on an eight-element set.

The layers of hierarchies A and B in Figure 6 are presented by dashed lines. The number of layers in tree B is smaller than in A because B is a complete

hierarchy. A binary hierarchy is referred to as a *complete* binary hierarchy if every interior cluster has children splitting it into parts of equal cardinality so that the total number of entities (pixels) is a power of two, $|I| = 2^n$, and each layer L^m has exactly 2^m classes consisting of 2^{n-m} elements each ($m = 0, 1, ..., n$). Obviously, for $|I| = 2^n$, the minimum number of layers, $n + 1$, is achieved only if S_W is complete (as B in Figure 6).

In the problems of data compression, the layers of a layered hierarchy can be exploited for approximate compression of the data. More specifically, with a layer $L_m = \{L_{mt}\}$ taken, a data vector $f = (f_i)$, $i \in I$, can be substituted by the vector of within class averages $f_{mt} = \sum_{i \in L_{mt}} f_i/|L_{mt}|$, which is considered as the data at m-th level of resolution. The smaller m, the coarser the resolution; the larger m, the finer the resolution. Taking into account the spatial character of the data, a different averaging operator can be employed, giving, say, smaller weights to entities which are farther from the middle.

The layers can be used also for recalculating the averages while running along the hierarchy bottom-to-up since, obviously,

$$nf_{mt} = n_1 f_{mt1} + n_2 f_{mt2}$$

where f_{mt1} and f_{mt2} are the averages within children of L_{mt}, and n, n_1, n_2 are cardinalities of L_{mt} and its respective children. The children obviously belong in L_{m+1}. It is not difficult also to exploit the hierarchy for recalculating the averages running up-down along the hierarchy. Let us save, for every cluster S_w, in addition to f_w, the between-split difference $d_w = f_{w1} - f_{w2}$, where f_{w1} and f_{w2} are the averages of f within respective children of S_w. The formulas

$$f_{w1} = f_w + \frac{n_{w2}}{n_w} d_w, \quad f_{w2} = f_w - \frac{n_{w1}}{n_w} d_w \tag{4.1}$$

provide for calculating the average values in L_{m+1} by the averages of L_m. This allows to make decompression of the data in a fast way: to recalculate all the averages starting from any upper layer, as for instance from the grand mean $f_0 = \sum_{i \in I} f_i/|I|$. The price for that: values d_w kept along the hierarchy. The cluster cardinalities kept is a part of "hard" information about the hierarchy; they do not depend on data. Formula (4.1) becomes especially simple for a complete binary hierarchy:

$$f_{w1} = f_w + d_w/2, \quad f_{w2} = f_w - d_w/2 \tag{4.2}$$

In Figure 7, the A and B hierarchies from Figure 6 are exploited for compressing a vector f whose values are the boxed digits: F version keeps all the averages, D all the differences. It can be seen that hierarchy A provides for a safe data compression: only one average, f_0 in F, and two differences, 1.6 and 1, are needed to decompress the data entirely: the other differences are zero and thus can be dropped out of consideration. This is because hierarchy A fits into data, f, better than B does.

This methodology can be put in the linear space framework as follows.

Let us consider a layered hierarchy $\mathcal{L}$ corresponding to a binary hierarchy S_W on I. Let us define, for any $L_m \in \mathcal{L}$ and $L_{mt} \in L_m$, normed binary indicator vector χ_{mt} where $\chi_{mt}(i) = 1/\sqrt{|L_{mt}|}$ if $i \in L_{mt}$ and $= 0$ otherwise. The χ vectors corresponding to different classes of L_m are, obviously, mutually orthogonal. Let us denote by V_m the subspace in $|I|$-dimensional space generated by the normed binary indicator vectors of m-th layer, L_m. Let us denote by D_m the subspace generated by the nest indicator vectors, $\phi_{mt}(i)$, of the nonsingleton classes $L_{mt} \in L_m$.

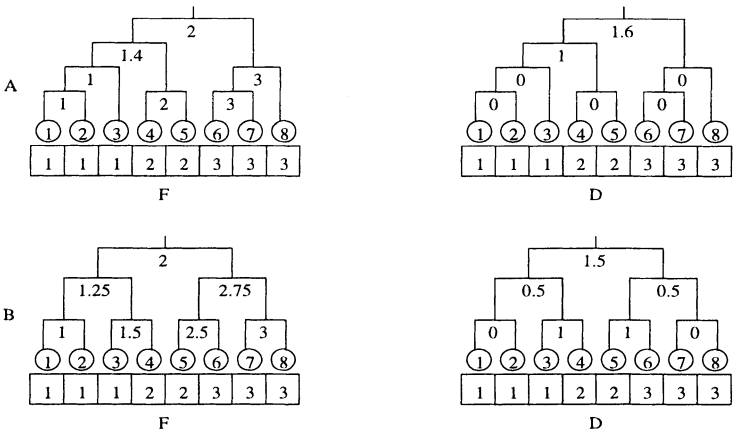


FIGURE 7. Compression and decompression of the boxed data with hierarchies A and B from Figure 6.

It is quite evident that the vectors χ_{mt} and ϕ_{mt} are pair-wise orthogonal, which implies that the spaces V_m and D_m are orthogonal too. Moreover, the following statement holds.

STATEMENT 6. *For any m ($m = 1, ..., n$), the subspace D_{m-1} is the orthogonal complement of V_{m-1} in V_m so that*

$$V_{m-1} \oplus D_{m-1} = V_m. \tag{4.3}$$

Consider a $|I|$-dimensional vector f projected into the subspace V_m:

$$f_i = \sum_t v_{mt}\chi_{mt}(i) + e_i \tag{4.4}$$

where e_i is the residual value. Due to equation (4.3), this can also be written as

$$f_i = \sum_t v_{m-1,t}\chi_{m-1,t}(i) + \sum_t c_{m-1,t}\phi_{m-1,t}(i) + e_i \tag{4.5}$$

with the same residuals.

The coefficients in (4.4) and (4.5) corresponding to a cluster $S_w \in S_W$ are: $v_w = f_w\sqrt{n_w}$ and $c_w = \sqrt{n_{w1}n_w/n_{w2}}(f_{w1} - f_w)$ where n_w, n_{w1}, n_{w2} are the cardinalities and f_w, f_{w1}, f_{w2} the within class averages for S_w and its children, S_{w1}, S_{w2}, respectively. The latter expression is the scalar product of f and ϕ_w and coincides with that in (2.9) while the former is equal to the scalar product of f and χ_w. These lead to the following formulas for fast recalculating the coefficient values along the hierarchy bottom-up:

$$v_w = v_{w1}\sqrt{n_{w1}/n_w} + v_{w2}\sqrt{n_{w2}/n_w}, \quad c_w = v_{w1}\sqrt{n_{w2}/n_w} - v_{w2}\sqrt{n_{w1}/n_w} \tag{4.6}$$

and up-down

$$v_{w1} = c_w\sqrt{n_{w2}/n_w} + v_w\sqrt{n_{w1}/n_w}, \quad v_{w2} = -c_w\sqrt{n_{w1}/n_w} + v_w\sqrt{n_{w2}/n_w} \tag{4.7}$$

These formulas are especially simple for complete hierarchies where all the coefficients in (4.6) and (4.7) become equal to $1/\sqrt{2}$.

4.2. Wavelets and Multiresolution Analysis. The contents of the previous section parallels some developments in data processing based on the so-called wavelet transformations. The concept of wavelet became quite popular immediately after it was introduced some ten years ago; it associates the most profound results of the theories of real-valued functions with the most urgent problems of image and other huge data compression and decompression (see, for example, reviews by Mallat (1989), Kay (1994), and Jawerth and Sweldens (1994)). The (discrete) wavelet theory involves two basic constructions: a multiresolution approximation of the space of all square-integrable real-valued functions L^2 and a dilation/translation family of functions $\chi_{mt} = 2^{m/2}\chi(2^m x - t)$ obtained from a so-called *scale* function $\chi(x)$ (which integrates to unity) with m "doubling" dilations of the space and translation of the origin by t. A basic function χ for the theory is the so-called box function $\chi(x) = \chi_{[0,1]}(x)$, that is, the indicator function of the interval $[0, 1]$ which is equal to 1 within the interval and 0 outside the interval.

The dilation/translation family may yield the functional approximation

$$f(x) = lim_{m\to\infty} \sum_t a_{mt}\chi_{mt}$$

to allow the sum $\sum_t a_{mt}\chi_{mt}$ to be considered as an approximation of any function $f \in L^2$ at resolution m for any fixed m. Here and below in this section, m and t are arbitrary integers.

A multiresolution approximation of L^2 is a sequence $\{V_m\}$ of closed subspaces of L^2 satisfying the following properties:

M1 $V_m \subset V_{m+1}$;
M2 The union of all V_ms is dense in L^2, and the intersection of them consists of 0 only;
M3 $f(x) \in V_m$ if and only if $f(2x) \in V_{m+1}$;
M4 $f(x) \in V_m \to f(x - 2^{-m}t) \in V_m$;
M5 V_0 is isomorphic to the set of all integer sequences that are square-summable.

The meaning of the properties are as follows: V_m are approximation subspaces which are nested, thus every finer resolution $m+1$ contains all the information necessary to find the coarser resolution m (M1); the approximation can be as complete or as rough as necessary (M2); every resolution level doubles the scale (M3, M4); there is a one-to-one correspondence between the representation of f at resolution m and the coefficients a_{mt} (M5).

Let us define the subspace D_m to be the orthogonal complement of V_m in V_{m+1}. Thus, it contains all the detail lost in moving from an approximation at the finer resolution $m + 1$ to the coarser resolution m, and satisfies the equality $V_{m+1} = V_m \oplus D_m$.

It appears, given a multiresolution approximation $\{V_m\}$, there exists a unique scaling function $\chi \in V_0$ and an associated $\phi \in D_0 = V_1 - V_0$ (called a wavelet) such that $\{\chi_{mt}\}$ forms an orthonormal basis for V_m and $\{\phi_{mt}\}$ forms an orthonormal basis for D_m. Thus, for any $f \in V_m$, there are two orthonormal decompositions holding:

$$f(x) = \sum_t a_{mt}\chi_{mt}(x) \tag{4.8}$$

and

$$f(x) = \sum_t a_{m-1,t}\chi_{m-1,t}(x) + \sum_t b_{m-1,t}\phi_{m-1,t}(x) \tag{4.9}$$

Decomposition (4.9) is interpreted as the reconstruction of a finer resolution involving both the coarser resolution decomposition and the "lost detail" decomposition through wavelets. The decompositions are obvious parallels to those in (4.4) and (4.5).

The pair of the scale and wavelet functions can be taken to satisfy the following equations:

$$\chi(x) = \sum_t c_t\chi(2x - t), \quad \phi(x) = \sum_t (-1)^t c_{1-t}\chi(2x - t) \tag{4.10}$$

which implies that the wavelet function corresponding to the box function is the so-called Haar wavelet $\phi(x)$ which is equal to 1 for $0 \leq x < 1/2$, -1 for $1/2 \leq x < 1$, and 0 for all other x.

Graphs of the box and Haar functions are shown in Figure 8.

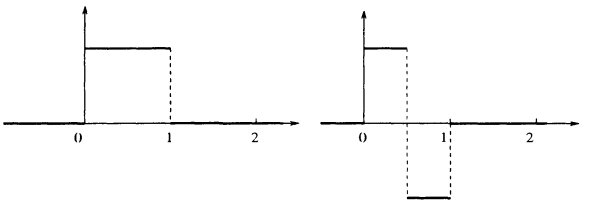


FIGURE 8. Graphs of the box and Haar functions.

The equations (4.8) to (4.10) are used to define the so-called fast wavelet transform allowing calculation of every coefficient at a finer resolution through the coefficients of a coarser resolution, and vice versa.

To apply these to image/signal processing, the following framework is employed. Let there be a pixellated unidimensional image at resolution m being a 2^m-dimensional vector v^m. This can be represented by a function $f(x) \in V_m$ defined as $f(x) = \sum_t v_t^m \chi_{mt}(x)$ where non-zero coefficients are from v^m. To calculate a coarser data sequence v^{m-1} which has half as many non-zero entries as v^m, the equations (4.8) and (4.9) are used; decompression of the data also can be done based on these equations. Moreover, the following holds.

STATEMENT 7. *The formulas (4.6) and (4.7) applied for a complete spatial binary hierarchy are a computational implementation of the fast wavelet transform based on the box scale and Haar wavelet functions.*

Sticking to the simplest box and wavelet functions restricts flexibility of the binary hierarchy approach. However, the discreteness of binary hierarchies makes up for that allowing compression and decompression of information without requiring any continuity or/and smoothness conditions which are mandatory in the classical case. Moreover, none of the "spatial" restrictions of the quantitative theories holds here: the hierarchy may be incomplete, the cluster cardinalities different, and the clusters may be spatially disconnected.

5. Extension onto Rectangle Objects

5.1. Bihierarchies and Quad-trees. The constructions above can be extended onto two-dimensional pixellated images via the following concept. A hierarchy S_W defined on $I = I' \times I''$ will be referred to as a *bihierarchy* if any of its clusters, S_w, is a Cartesian rectangle, that is, $S_w = A \times B$ for some $A \subseteq I'$ and $B \subseteq I''$, and the children of S_w are $A1 \times B1$, $A1 \times B2$, $A2 \times B1$, and $A2 \times B2$ for some partitions, $\{A1, A2\}$ and $\{B1, B2\}$, of A and B, respectively. (To allow more freedom in handling "one-dimensional" strip clusters, $\{i'\} \times B$ or $A \times \{i''\}$, we can admit some of the subsets as being empty.) The sets, A and B, can be referred to as the ranges of S_w in I' and I'', respectively. A bihierarchy will be called *spatial* if I' and I'' are ordered and the ranges of all clusters are intervals of these orders. A specific case of a bihierarchy is the Cartesian product of two binary hierarchies, $S_W = S'_{W'} \times S''_{W''}$, the clusters of which are all possible Cartesian products of clusters of $S'_{W'}$ and $S''_{W''}$.

A (divisive) bihierarchical cluster structure is an "upper" part of a bihierarchy (defined by relaxing the condition that every singleton $(i', i'') \in I' \times I''$ belongs to the bihierarchy).

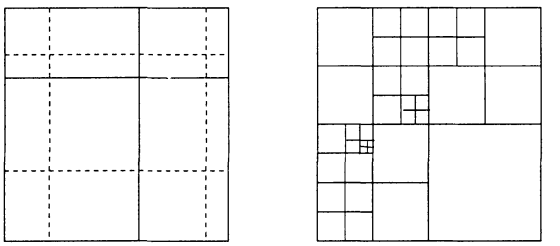


FIGURE 9. Higher splits of a Cartesian product of two spatial binary hierarchies (a) and a quad-tree (b).

A well-known structure in image data analysis, the quad-tree (see, for example, Burt and Adelson, 1983, Samet, 1990) fits into the concepts introduced. In our terms, a quadtree is a bihierarchical cluster structure for a complete spatial bihierarchy (see Figure 9, (b)).

For a cluster S_w in a bihierarchy, S_W, with its ranges A and B subdivided in $A1, A2$ and $B1, B2$, respectively, three nest indicator functions are needed according to the general description in section 2.3. A natural way of defining the indicators would be by considering the four children as produced via double dichotomy. In such a double dichotomy cluster $S_w = A \times B$ can be divided, firstly, in two strips, say, $A1 \times B$ and $A2 \times B$, and secondly, each of the strips is further split into the final children $Ak \times Bj$, $k, j = 1, 2$. The three splits can be assigned with corresponding nest indicator functions. The bihierarchy can be regarded as a contracted version of the binary hierarchy involving the double dichotomy described.

However, we'll consider here another triple of nest indicator functions (also different from those defined in section 2.3). Each of the ranges implies its nest indicator function, $\phi_A(i')$ and $\phi_B(i'')$, defined with correspondingly modified formulas (2.1) and (2.2). The three cluster nest indicator functions, ϕ_A, ϕ_B, and ϕ_{AB}, then, can be defined for all $(i', i'') \in S_w = A \times B$ as (1) $\phi_A(i', i'') = \phi_A(i')\chi_B(i'')$,

(2) $\phi_B(i',i'') = \chi_A(i')\phi_B(i'')$, and (3) $\phi_{AB}(i',i'') = \phi_A(i')\phi_B(i'')$ where $\chi_S(i) = 1/\sqrt{(|S|)}$ when $i \in S$ and $= 0$ when $i \notin S$. (When A or B is a singleton, only one of these three functions remains valid.) These functions, obviously, are centered and normed (with regard to all $(i',i'') \in I' \times I''$) and, moreover, are mutually orthogonal. Thus, the nest indicator functions of all interior clusters $S_w \in S_W$ form an orthonormal basis, Φ, of the space of $|I' \times I''|$-dimensional centered matrices (considered as vectors). The coefficients of decomposition of a matrix vector $y(i',i'')$ defined on $I' \times I''$ by the fragment of Φ related to a cluster $S_w = S_{AB}$ are scalar products of $y(i',i'')$ and corresponding nest indicator functions that can be shown to have the following format:

$$c_A = \sqrt{\frac{n_{A1}n_{A2}}{n_A}}\sqrt{n_B}(y_{1.} - y_{2.}),$$

$$c_B = \sqrt{n_A}\sqrt{\frac{n_{B1}n_{B2}}{n_B}}(y_{.1} - y_{.2}) \tag{5.1}$$

$$c_{AB} = \sqrt{\frac{n_{A1}n_{A2}}{n_A}}\sqrt{\frac{n_{B1}n_{B2}}{n_B}}(y_{11} - y_{12} - y_{21} + y_{22})$$

where y_{kj}, $y_{k.}$, or $y_{.j}$ is the average of $y(i',i'')$ on $Ak \times Bj$, $Ak \times B$ or $A \times Bj$, respectively $(k,j = 1,2)$.

These expressions can be easily extended to the situation of three-way data $Y = (y(i',i'',k))$ by adding an index k where necessary.

Applications to analysis of rectangle data can be done by extending the developments above to bihierarchies.

5.2. Bihierarchical Clustering. Following the sequential extraction strategy SEFIT discussed in section 3, we arrive at the problem of splitting the ranges of a given rectangle $A \times B \subseteq I' \times I''$ to maximize $\mu^2_{AB} = c^2_A + c^2_B + c^2_{AB}$ where the items are defined in (5.1):

$$\mu^2_{AB} = \frac{n_{A1}n_{A2}}{n_A}\frac{n_{B1}n_{B2}}{n_B}(y_{11} - y_{12} - y_{21} + y_{22})^2 + \frac{n_{A1}n_{A2}}{n_A}n_B(y_{1.} - y_{2.})^2 + n_A\frac{n_{B1}n_{B2}}{n_B}(y_{.1} - y_{.2})^2 \tag{5.2}$$

This can be done with a local search algorithm. For instance, to find an initial partition, let us split A to maximize c^2_A and, in parallel, B to maximize c^2_B. This can be done with an algorithm for splitting a cluster described in section 3.2. Then, the partition found can be iteratively updated by exchanging rows between $A1$ and $A2$ or columns between $B1$ and $B2$ (one item in a time) until μ^2_{AB} cannot be increased anymore.

5.3. Up-to-Bottom Decompression. A bihierarchy can be employed for data compression and decompression in the same fashion as it was described above for hierarchies. We will not maintain here the linear subspace terminology since it does not much differ from that described above. Let us just show how the data compressed as within cluster averages can be decompressed up-down employing the

three differences involved in (5.1) and kept as coefficients of the "wavelet" bases consisting of those parts of Φ that correspond to layers of a bihierarchy S_W:

$$d_{AB} = y_{11} - y_{12} - y_{21} + y_{22},\ d_A = y_{1.} - y_{2.},\ d_B = y_{.1} - y_{.2}.$$

STATEMENT 8. *In a bihierarchy, the children's averages can be expressed through the within cluster S_w average, y_w, and the d-coefficients above as follows:*

$$y_{11} = y_w + \frac{n_{A2}}{n_A}\frac{n_{B2}}{n_B}d_{AB} + \frac{n_{A2}}{n_A}d_A + \frac{n_{B2}}{n_B}d_B,$$

$$y_{12} = y_w - \frac{n_{A2}}{n_A}\frac{n_{B1}}{n_B}d_{AB} + \frac{n_{A2}}{n_A}d_A - \frac{n_{B1}}{n_B}d_B,$$

$$y_{21} = y_w - \frac{n_{A1}}{n_A}\frac{n_{B2}}{n_B}d_{AB} - \frac{n_{A1}}{n_A}d_A + \frac{n_{B2}}{n_B}d_B,$$

$$y_{22} = y_w + \frac{n_{A1}}{n_A}\frac{n_{B1}}{n_B}d_{AB} - \frac{n_{A1}}{n_A}d_A - \frac{n_{B1}}{n_B}d_B.$$

Proof: The proof follows with a little arithmetic from the basic equations connecting y_w, $y_{k.}$ and $y_{.j}$ with y_{kj}, $k,j = 1,2$, as, for instance $n_A n_B y_w = n_{A1}n_{B1}y_{11} + n_{A1}n_{B2}y_{12} + n_{A2}n_{B1}y_{21} + n_{A2}n_{B2}y_{22}$, and definitions of d_{AB}, d_A, d_B. □

These formulas can be converted into the language of V_m and D_m spaces as it was done in the case of hierarchies.

6. Conclusion

The following issues discussed in the paper seem of an interest:

1. Every binary cluster hierarchy is associated with an orthonormal basis of the centered variable space providing a SVD-like decomposition of the data matrix by the elements of the cluster structure.
2. The set of interpretation aids based on the SVD-like decomposition adds the decompositions of the single variable variances, variable-to-variable covariances/correlations, and entity-to-variable entries by the clusters to the known decomposition of the overall variance.
3. An existing divisive clustering strategy can be explained as a "greedy" one-by-one fitting strategy for a clustering approximation model in terms of the SVD-like decomposition.
4. Norming of the data with norms which are different from the Euclidean one (like Chebyshev's norm related to the range of a variable rather than to its density) might lead to better clustering results because of a natural ordering of the variables emerging.
5. The binary hierarchies, applied to spatial data processing, are very closely related to wavelets and quadtrees which correspond to the so-called complete (spatially and numerically) hierarchies and bihierarchies, respectively.
6. Bottom-up and up-down computations along the binary hierarchies are parallel to the so-called fast wavelet transforms, and can be used in data compression/decompression problems.
7. The discrete character of binary hierarchies allows relaxing many restrictions of the wavelet-based techniques since the hierarchy clusters may be split into parts which are neither of equal sizes nor spatially continuous. Still, fast recalculation formulas hold for such general hierarchies and bihierarchies, which should be exploited in data processing.

8. Combining hierarchy-based clustering with the follow-up data processing may be an adequate tool for processing sets of data that have a steady structure (as documents of a kind or images of a body organ).

7. Acknowledgement

The author thanks F.R. McMorris for his numerous revising suggestions.

References

[1] Braverman, E.M., and Muchnik, I.B. (1983) *Structural Methods for Processing Empirical Data*, Moscow: Nauka (in Russian).

[2] Breiman, L., Friedman, J.H., Olshen, R.A., and Stone, C.J. (1984) *Classification and Regression Trees*, Belmont, Ca: Wadswarth International Group.

[3] Burt, P.J., and Adelson, E.H. (1983) "The Laplacian pyramid as a compact image code", *IEEE Transactions on Communications V COM-31*, 532-540.

[4] Chaturvedi, A., and Carroll, J.D. (1994) "An alternating optimization approach to fitting INDCLUS and generalized INDCLUS models", *Journal of Classification, 11*, 155-170.

[5] Diday, E. (1986) "Orders and overlapping clusters by pyramids", In: J. de Leeuw, W. Heiser, J. Meulman, and F. Critchley (Eds.) *Multidimensional Data Analysis*, Leiden: DSWO Press, 201-234.

[6] Edwards, A.W.F. and Cavalli-Sforza, L.L. (1965) "A method for cluster analysis", *Biometrics, 21*, 362-375.

[7] Golub, G.H. and Van Loan, C.F. (1989) *Matrix Computations*, Baltimore: J. Hopkins University Press.

[8] Gower, J.C. (1967) "A comparison of some methods of cluster analysis", *Biometrics, 23*, 623-637.

[9] Hansen, P., Jaumard, B., and Da Silva, E. (1993) "Average-linkage divisive hierarchical clustering", *Les Cahiers du GERAD, G-91-55*, Montréal.

[10] Hartigan, J.A. (1972) "Direct Clustering of a Data Matrix", *Journal of American Statistical Association, 67*, 123–129.

[11] Jain, A.K. and Dubes, R.C. (1988) *Algorithms for Clustering Data*, Englewood Cliffs, NJ: Prentice Hall.

[12] Jolliffe, I. T. (1986) *Principal Component Analysis*. New York: Springer-Verlag.

[13] Kay, J. (1994) "Wavelets", In *Advances in Applied Statistics, 2*, 209-224.

[14] Lance, G.N., and Williams, W.T. (1967) "A general theory of classificatory sorting strategies: 1. Hierarchical Systems", *Comp. Journal, 9*, 373-380.

[15] Mallat, S.G. (1989) "Multiresolution approximations and wavelet orthonormal bases on $L^2(R)$", *Transactions of the American Mathematical Society, 315*, 69-87.

[16] Milligan, G.W., and Cooper, M.C. (1988) "A study of standardization of the variables in cluster analysis", *Journal of Classification, 5*, 181-204.

[17] Mirkin, B.G. (1990) "A sequential fitting procedure for linear data analysis models", *Journal of Classification*, 7, 167-195.

[18] Mirkin, B.G. (1994) "Approximation of association data by structures and clusters". In: P. Pardalos, H. Wolkowicz (Eds.) *Quadratic Assignement and Related Problems*, DIMACS Series v. 16, American Mathematical Society, 293-316.

[19] Mirkin, B.G., Arabie, P., and Hubert, L. (1995) "Additive two-mode clustering: The error-variance approach revisited", *Journal of Classification, 12*, 243-263.

[20] Rosenberg, S. (1982) "The method of sorting in multivariate research with applications selected from cognitive psychology and person perception", In N. Hirchberg and L.G. Humphreys (Eds.) *Multivariate Applications in the Social Sciences*, University of Illinois at Urbana–Champaign: L. Erlbaum Assoc., 117 - 142.

[21] Samet, H. (1990) *The Design and Analysis of Spatial Data Structures*. Addison-Wesley Series on Computer Science and Information Processing. Addison-Wesley Publishing Company.

[22] Shepard, R. N., and Arabie, P. (1979) "Additive clustering: representation of similarities as combinations of discrete overlapping properties", *Psychological Review, 86*, 87-123.

[23] Vingron, M., and Sibbald, P.R. (1993) "Weighting in sequence space: a comparison of methods in terms of generalized sequences", *Proc. Natl. Acad. Sci. USA, 90*, 8777-8781.

[24] Ward, J.H., Jr (1963) "Hierarchical grouping to optimize an objective function", *Journal of American Statist. Assoc., 58*, 236-244.

DIMACS, RUTGERS UNIVERSITY, P.O.BOX 1179, PISCATAWAY, NJ 08855-1179 USA.
E-mail address: mirkin@dimacs.rutgers.edu

https://doi.org/10.1090/dimacs/037/24

DIMACS Series in Discrete Mathematics
and Theoretical Computer Science
Volume **37**, 1997

Learning Algorithms Generating Multigranular Hierarchies

ALEX MEYSTEL

ABSTRACT. This paper introduces an approach for analysis of systems with learning, including living systems and artificial intelligent systems. An architecture of an automaton with learning and behavior generation is presented which determines a class of learning algorithms. An algorithm of unsupervised learning that employs grouping, focusing attention and combinatorial search is described. Top down computational processes with growth of the multigranular hierarchy of knowledge acquired are promulgated. The joint system of Behavior Generation and Learning searches for a preferable motion trajectory. It creates top down planning/control processes leading to the temporal evolution of the system. An argument is presented for a joint analysis of spatial and temporal processes of behavior and learning.

1. Introduction

In 1986 a concept of "baby-robot" was introduced by the author [1]. This concept emphasizes processes of unsupervised learning and is associated with the joint evolution of behavior and knowledge incorporated within a system. It seems plausible that learning from multiple experiences is inseparable from the architecture of applying the acquired knowledge for shaping the desirable behavior.

Analysis of the evolution of living creatures, problems of knowledge acquisition, mechanisms of intelligence which is a result of the evolution, all survival oriented matters are beyond the discussion in the paper. Instead of analyzing living creatures, we

---

Key words and phrases: complexity, control, evolution in biology, experiences, focusing attention, generalization, granularity, grouping, hierarchical systems, intelligence, multiresolutional architecture, planning, unsupervised learning.

introduce a formal system—a learning automaton.

It has the faculties required to obtain and use knowledge: sensors, subsystems for storing and organizing information, subsystems for generating commands, actuators for changing the world, but initially it has no world model and no rules to achieve a goal. This knowledge should be learned and our goal to understand how.

The concepts of *experience* as a part of *behavior,* as well as desirability of behavior and its components-experiences, will be introduced and explored (Section 3). Behavior is shaped by the *actions* which emerge as a result of planning the goal-states, actions that lead to goal-states, and strings of them. The process of finding the string of desirable states and actions leading to them is called *planning.* The latter would be impossible unless the ability to judge the degree of desirability of states and actions would not be acquired in advance. Generating and applying proper commands to execute the planned trajectory and compensate for errors is called *execution.* Both planning and execution is a part of *control.* The process of acquiring the relevant information and processing it so that a proper behavior could be generated is called *learning.* Planning/control and learning are complementary computational procedures, merging into a joint process of intelligent computation.

The process of planning starts with *focusing attention* which selects the initial representation of the world or the map with its boundaries. Maps are discretized into *tessellata* which determine maps' resolution, or granularity. *Combinatorial search* is performed as a procedure of choosing one string (the minimum cost string) out of the multiplicity of all possible strings formed out of the space tessellata at this particular level of *resolution.*

Grouping the tessellata in a variety of feasible strings allows selection of one of them. This is necessary for the subsequent evolution of the behavior grouping, which is followed by generalization that constructs an envelope around the vicinity of the minimum cost string. This envelope is being submitted to the next level of resolution where the next cycle of computation starts. Focusing attention presumes proper distribution of nodes in the state space so that no unnecessary search be performed. Combinatorial search forms the alternatives. All of these three procedures together can be considered as a process of generalization. Generalization is a generation of the map for the subsequent search at the lower level of resolution.

An example of joint functioning of these three operations is shown in Figure 1. The operations work jointly as a triplet, which can be considered an *elementary unit of intelligence* [**2**].

This process can proceed in the opposite direction, from a lower resolution to a higher resolution, i. e. counterclockwise. This is contrary to the process is shown in Figure 1. However, it won't be a generalization anymore. It will be an opposite process of instantiation. Instead of *grouping,* it employs *decomposition.* Both generalization and instantiation are the key processes of intelligence. In this light, the consecutive sections of this paper should be viewed.

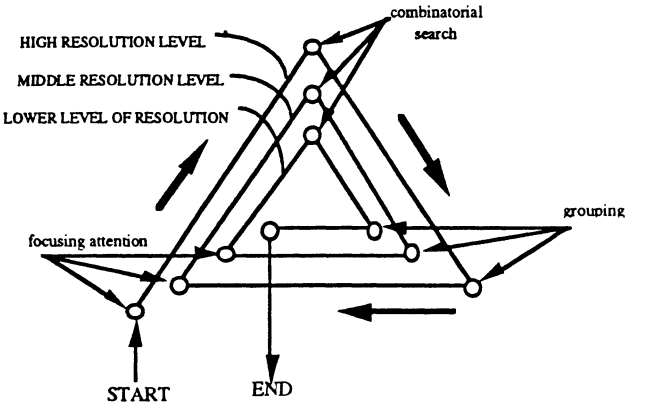


Figure 1. Conceptual Structure of the Process of Multiresolutional Consecutive Knowledge Acquisition by the virtue of Generalization

All of these phenomena presume that the system under consideration can "behave" as a result of its planning decisions and actions of the actuators. It is required that the changes in the world are reported to the system by a set of sensors. The overall system can be represented using the concept of elementary loop of functioning [2-4].

A new type of unsupervised learning, with combinatorial enhancement and generalization, is explored in this paper. It is demonstrated that this learning leads to emergence of a multiresolutional representation and a hierarchy of control. This type of learning system provides an explanation of evolutionary processes. Evolution of knowledge is discussed. It is considered that the evolution of living creatures is a special case of the evolution of knowledge representation.

The process of evolution of the learning automaton can be characterized by a gradual sophistication of its knowledge and behavior. This gradualness is not monotonic in all derivatives. The evolution of knowledge is rather "punctuated." It develops in steps that emerge when a collection of hypotheses gets transformed into a rule. The evolution of behavior has similar steps due to the *emergence* of new skills which happens also in different moments in time. Our model is intended to be used for the processes both of

early and mature learning. In final Sections, a unified process of learning and behavior generation is discussed.

2. Algorithms of Unsupervised Learning with Combinatorial Enhancement and Generalization

2.1 Preliminaries. An automaton is a state machine which generates outputs by using its transition function and state-output function tabulated in advance for all possible situations. A class of learning automata is introduced which are state machines whose transition and state-output functions are updated and modified based upon prior experiences which are stored in the memory and transformed into sets of rules. The transition and output functions of these automata have open lists of rules. New rules can be added to these lists. The aim of a learning automaton is to acquire rules which will increase the total reward to achieve the goal, or to increase the value of an objective criterion of optimality associated with the goal achievement. This concept is similar to the one described in [5].

Thus, learning automata are presumed to have a learning system L which allows for enriching both the transition and the output functions. We will demonstrate that as the system of rules develops it becomes a hierarchical one. This includes the corresponding input and output vocabularies (V_I *and* V_O.). This is equivalent to formation of the hierarchy of automata as a result of the evolution of a single learning automaton. Operators of L are equipped by minimum initial set of tools, or "bootstrap knowledge." This includes the ability to form strings, to construct hypothetical implications, and to infer tautologies.

Learning system L can be defined as a system of acquisition of experiences, transformation of these experiences into rules of action, derivation of new concepts, and organization of these concepts into knowledge and decision-making structures suitable for achieving the goal.

In order to support processes of learning, each learning automaton is equipped by the set of actuators that follow commands at the output of learning automaton. Changes in the world are measured by the sensors. A set of sensors is the only source of information for the learning system about the state of the World (Situation.) The automaton is also equipped by subsystems of Sensory Processing and World Model which allow for interpreting the input from sensors. All these modules are discussed as a discrete event system.

The Learning System can judge upon truthfulness of this information only by the

results of actions (behavior) which are undertaken to achieve the goal. The subsystem of Behavior Generation (BG) contains *transition function* and *output function* [2, 3, 6]. Other devices of BG are described in Section 5. This section is a further development of the approach presented in [7, 8].

We use the term *situation* to describe the state of the world represented as a set of n sensors' outputs which arrive at a moment of time i

$$S_i = \{s_{ji}\}, \quad j = 1,...,n; \ i=1,...T \tag{1}$$

where s_{ji} is a numerical value for the j sensor.

Action is a set of m action outputs which are generated by the system at moment of time i

$$A_{i,i+1} = \{a_{ji}\}, \quad j = 1,...,m; \ i=1,...T \tag{2}$$

where a_{ji} is a numerical value that has been observed as an output of actuator j.

Situations can be represented as sets (lists), or as vectors within a particular system of coordinates. Actions are understood as causes of changes that are sensed after the actions are applied. This allows for recording correlations between these causes and the sensed changes which introduces the concept of a rule.

We use the term *experience* instead of cause-effect relationship because we want to underscore the fact that no prior cause-effect knowledge is available. Sometimes, the functioning of learning automata is described in terms of the quadruplets: states-before-action, states-after-action, actions, rewards. For example, a "single move" experience can be described as follows:

$$E_{i,i+1} = \{S_i, A_{i,i+1}, S_{i+1}, J_{i,i+1}(G)\}, \ i=1,2,...T; \tag{3}$$

where S_i is a vector of *situation* at the time i, $A_{i,i+1}$ is a vector of *action* applied to S_i, S_{i+1} is a vector of the next situation in which $A_{i,i+1}$ has ended. In dynamic systems, we are interested in longer strings of moves which we regard as experiences. Any concatenation of single moves generates a multi-move experience of the type "situation-action-situation-...-situation" which serve a basis for the subsequent learning.

G is a *goal.* It is a situation which must be achieved as a result of the behavior. The goal is often presumed to be given to a system. From [2-6] we know that goals can emerge also as a result of planning. From [3,4] we know that the process of learning

generates subgoals.

J(G) is a *goodness* or *reward* attained under the goal G as a result of a single move. It is presumed that any valued experience is associated with a certain measure of "goodness" $J_{i,i+j}(G)$. It can be interpreted as a reward for the pursuit of the *goal G.* In a multi-move experience the value of reward is determined only by the initial and final situations. Both single-move and multi-move experiences will be regarded as unit of experiences.

The value of reward will be used for the subsequent process of hypotheses generation and selection. Ultimately, it determines the rules necessary for "survival." A *rule* can be obtained as a result of transforming the cause-effect relations discovered within repetitive experiences.

There are two forms of rules:

Control Rules (CR): IF present situation & goal —> THEN action required

and

Event Rules (ER): IF present situation & action —> THEN new situation

To find new rules an operation of generalization should be applied. In subsection 2.3 the operation of Inductive Generalization is described. Clusters of similar experiences should be created. Then, the control rules and event rules can be generated as hypotheses. These hypotheses are used by an automaton as rules for behavior generation. A frequency of valuable rewards estimates plausibility of a newly created rule and confirms its correctness if the frequency is high.

After the rules are confirmed, some of their parts which emerged as a result of generalization, may not belong to the initial vocabularies *(V_I and V_O.)* These parts are to be treated as new *concepts. Concept* is a label for the entity which can be obtained recursively as a cluster or group of higher resolution concepts. All new concepts are obtained from two available classes of rules (see Figure 2.)

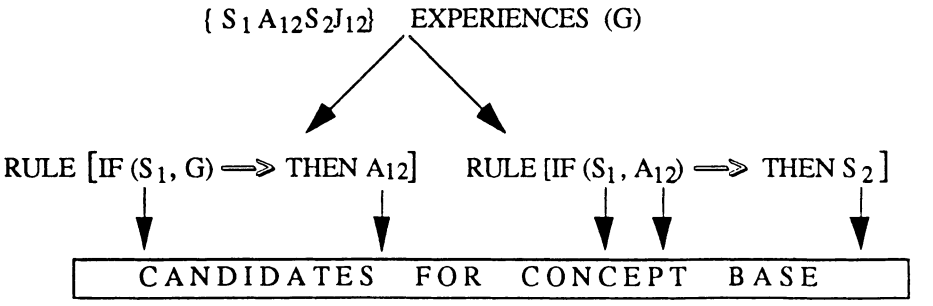


Figure 2. Experiences—>Rules—>Concepts

Situations and actions are concepts and so are clusters of situations and actions, as well as their components. They are organized into "ontology" base which stores them together with their relationships.

The new concept obtained from the generalized experiences as a result of their transformation into rules are *novel words* which are not present in the lists of previously defined input and output vocabularies. These novel words are words of the new vocabularies: at the lower level of resolution. They describe the *phenomena* related to groups of units of experiences.

Thus, the single sensor value becomes a group of sensor values which are unified by the fact that a particular action or string of actions lead to highly rewarded results.

Goodness or *reward* (J) is computed in a different way depending on a particular application. For example, it can be computed as a difference between increments of cost $C_{i,i+1}$ accumulated for the interval from state i to state (i+1), and $C_{i-1,i}$ accumulated for the interval from state (i-1) to state i.

The following sequence of activities can be explicated from the definition of learning:

1. Experiences $\{E_i\}$ are collected and stored in memory.
2. Experiences are compared, resemblances are determined, and clusters are formed by the virtue of resemblance.
3. Clusters of experiences which already have cause-effect relationships, are transformed into rules, and control rules and events rules are separated.
4. The clusters of situations and actions that are parts of the newly created rules are stored as concepts of lower resolution; then the growth of the concept base begins. New concepts emerge as a result of clustering.These concepts are labeled and receive a status of a new word.
5. The words for the clusters form a vocabulary. This vocabulary is regarded as a lower resolution vocabulary. Thus, in addition to the previous vocabularies V_I and V_O we obtained two new vocabularies V_I and V_O.
6. Subsequently, the new experiences are stored in parallel in two new vocabularies: the original one V_I and V_O, and the one formed by the newly created concepts V_I and V_O.

The process of consecutive operations {collection of experience—>hypothesis formation—>generation of rules—>concepts emergence} is repeated each time as a new experience arrives. As vocabularies V_I and V_O grow, they allow to represent and control

functioning of the system by using their new words. Thus, new, generalized experiences can be recorded for generalized situations and actions. The sequence of steps 1-6 can be repeated which will result in building up a new lower level of representation.

This algorithm describes a recursive process which leads to a multiresolutional system of world representation and a multiresolutional systems of rules of actions, both acquired as a result of learning.

The described learning process employs only *focusing attention* and *grouping* the experiences by their existing features of resemblance. This is not always satisfactory for successful learning. The third component is required from the triplet shown in Figure 1. This component is *combinatorial search.* It will be introduced in a form of combinatorial enhancement.

Learning with generalization allows for using experiences in a very efficient way. Multiple clustered experiences are treated as group phenomena. Rules generated by them are group rules. When the number of group rules becomes large, the algorithm of generalization can be applied again. The group rules of even lower resolution emerge.

However, this consecutive generalization is just a tool of reducing complexity. In order to receive innovations, some combinatorics should be introduced for generating words beyond the existing experiences. Similar mechanism has been proven to be very useful for design purposes [**10**].

2.2 Combinatorial Enhancement: Searching for Hidden Implications.

Each experience can testify only about some part of the overall situation. This is why in the theory of learning, the concept of a *combinatorial enhancement* which produces enhanced actions and situations has been introduced.

Indeed, the available sensor information not necessarily can be a good basis for generating a hypothesis. Consider an example with autonomous mobile robot which must learn how to act in a particular environment. In a particular situation, a single actuator command cannot be a proper response. Frequently, a combined command (steering + propulsion) must be assigned. Just value of the sensor reading of angle of "heading", $\angle\alpha$, or value of "angle to the goal", $\angle\sigma$, cannot be an antecedent in a rule what to do. However, their difference ($\angle\alpha - \angle\sigma$) is a perfect variable of control.

Such two persuasive experiences of learning as the evolution of living creatures and the evolution of knowledge would be substantially impaired if no combinatorial enhancement would be possible.

The following factors are considered critical for stimulating evolution of living

creatures: reproduction, mutation, competition, selection [9]. These factors are applied to knowledge in a similar way. Reproduction is like formation of statements of experiences. Competition and selection are like generalization of the best results with the highest values of rewards (see Subsection 2.3.) Search in a form of combinatorial enhancement is the direct analog of mutations. It can be compared to crossover and other mechanisms of forming new entities with an element of randomization.

We call formation of *combinations* of the available data for the consecutive searching and grouping *combinatorial enhancement*. For example, if measurable values v_1 and v_2 are known in the beginning of some experience, neither v_1, nor v_2 might be indicative of the need to use a particular action while the value of $(v_2 - v_1)$, or $(v_2 + v_1)$, or other combinations might be important to interpret results of measurements.

The enhancing takes as input a set of values and gives as output a combination among these values:

$$(4) \qquad Comb\ \{ v_1, v_2, v_3, \ldots, v_n \} = \{ v_i , (v_i \pm v_j) \}, \quad i=1\ldots n,\ j=1,\ldots n,\ i \neq j$$

This operation creates a new set containing the initial set and all possible combinations of the elements of the initial set. In (4) only combinations of additions and subtractions are illustrated.

An *enhanced representation* of a situation is constructed by considering enhanced vectors of situations and actions applied. For example, if the vector of situation was built upon readings of three sensors $S=\{s_1, s_2, s_3\}$, the enhanced vector of situation will include nine coordinates $\{s_1, s_2, s_3, (s_1+s_2), (s_1+s_3), (s_2+s_3), (s_1-s_2), (s_1-s_3), (s_2-s_3)\}$. Similar change will happen to the vector of actions: instead of $A=\{a_1, a_2, a_3\}$ a new vector will emerge: $\{a_1, a_2, a_3, (a_1+a_2), (a_1+a_3), (a_2+a_3), (a_1-a_2), (a_1-a_3), (a_2-a_3)\}$. It would be prudent to consider a new hybrid vector $\{(a_1+s_1), (a_1+s_1), (a_1+s_3), (a_2+s_1), (a_2+s_2), \ldots, (a_2-s_3), (a_3-s_1), (a_3-s_2), (a_3-s_3)\}$. However, in the simple examples that we used from the area of robotics, it was hard to come up with physical interpretation of each hybrid combination.

Thus, the *enhanced situation* is a set formed by a situation, all possible combination of its components, and all possible combinations between components of the action A and the situation S. Each experience presented by (3), or a string of single-move experiences (3) should be stored together with the synthesized *enhanced situation* S_{Enh} and *enhanced action* A_{Enh}.

2.3 The Algorithm of Inductive Generalization. The process of *inductive generalization* is one of the possible procedures of generalization. In addition to formation of clusters typical for any generalization, inductive generalization uses information about dynamic properties of the statistics of cluster formation. The maxim of inductive generalization claims that multiple occurrences of similar experiences testify for existence of a particular rule, if most of these occurrences have the same (or similar) explanation of causes [12]. If the number of occurrences is not statistically persuasive, then we can talk about the case of *hypotheses generation* by means of *abductive generalization.* In both cases, it is important to account for the list of attributes/variables of the situation, and also for the relations among them [13].

The process of unsupervised learning employs the algorithm of generalization which presumes a multiple iteration of the triplet from Figure 1: *focusing attention* on the subset of experiences and *searching* among them and their combinations until a *grouping* can be performed, i.e. a cluster of similar units could be substituted by a single generalized hypothesis [10]. This hypothesis is then stored in the database of hypotheses. If the subsequent experiences confirm the hypothesis, it becomes stronger.

This is not a trivial operation. Automata with operation of generalization have not been previously discussed in the literature on learning. The operation of generalization includes steps 1-4 of the algorithm of learning.

Thus, the stored experiences and the goal both are inputs to the process of generalization only in the very beginning. Then, the third input to the process of generalization not included into Figure 3. is information from the database of hypotheses for behavior generation.

This database is useful for the algorithm of generalization, and has two major functions:

• it allows for restoring information that was already lost in erased experiences; the hypotheses included in the database are generalizations (or compressions) of these lost units of information.

• it does not allow for creation of hypotheses which were already created, or creating hypotheses that have been already created earlier but demonstrated to be not valid.

Since the algorithm of learning as applied recursively to its own results, each two adjacent levels of resolution use generalization applied to the initial information which can be considered experiences and to generalized information which is called hypotheses. Thus, we can distinguish two kinds of generalization pertaining to the level to which they are applied: generalization from experiences before any hypothesis is available and generalization from hypotheses.

The previously created hypotheses supply the goal for generalization of experiences. If the database of hypotheses is empty the main goal of functioning is applied.

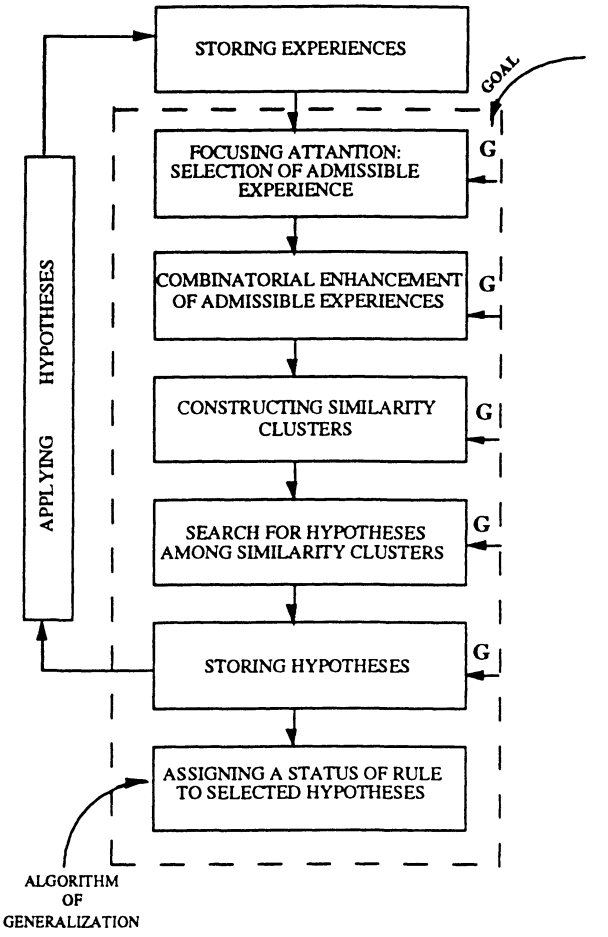


Figure 3. The Generalization Algorithm

The following factors are characteristic for the process of generalization and should be taken in account in any concrete computer algorithm. They are applicable both for generalizing experiences and hypotheses. We may treat hypotheses as experiences by themselves. The hypotheses are validated as a result of applying them. Thus, they become subjects for the subsequent generalization which is characterized by the following factors:

1. The most straightforward method of clustering is based upon extraction from the data base of two important subsets, admissible and non-admissible. Admissible experiences are "good," while non-admissible are "bad" with respect to the given goal which is determined by values of goodness/reward they deliver. The degree of goodness

which determines the threshold of clustering can vary. It will determine the productivity of learning and eventually the success of functioning.

2. The admissible set is used to generates hypotheses of rules that prescribe "what to do", while the non-admissible set generates hypotheses with a warning content: "this should not be done" in this situation.

3. The representation of each experience is combinatorially enhanced. Actually, enhancement plays the role of "imagination" in the overall process of generalization. It is similar to creating the strings of anticipated trajectories in the subsystem of Behavior Generation (Section 3).

4. Searching is performed to determine groups of similarity among the enhanced experiences, and the clusters are created.

5. Each of the clusters of similar enhanced experiences is considered to be a candidate for becoming a rule hypothesis.

6. Experiences are tested, and the results of testing enter the base of experiences.

7. Hypotheses are validated by statistics of their use and then enter the base of hypotheses.

All manipulations with sensor/actuator values are performed after the values are normalized. The reason for normalization comes from the fact that different sensor and actuators work over different ranges. Thus, *normalization* is a function which maps a value from a particular interval into the normalized value interval [0–1]. *De-normalization* is the inverse function of normalization. *Re-normalization* is applied when a new value arrives which is outside of the existing interval. It is necessary to re-normalize all the values previously normalized with the old interval.

The algorithm of generalization creates new hypotheses, rules, and concepts which become new words in vocabularies. The same algorithm is applied to its own results. Now the previously generalized experiences are generalized again, and the hypotheses and rules of "lower resolution" are obtained. This evolution of acquired knowledge is illustrated in Figure 4.

This diagram shows that the joint system of knowledge representation and behavior generation converges to a multiresolutional one. This convergence is completed, only after many new concepts are obtained and the consecutive generalization processes run a sufficient number of times. Then, formation of levels can be obtained by clustering of the multiplicity of outcomes of these consecutive generalizations.

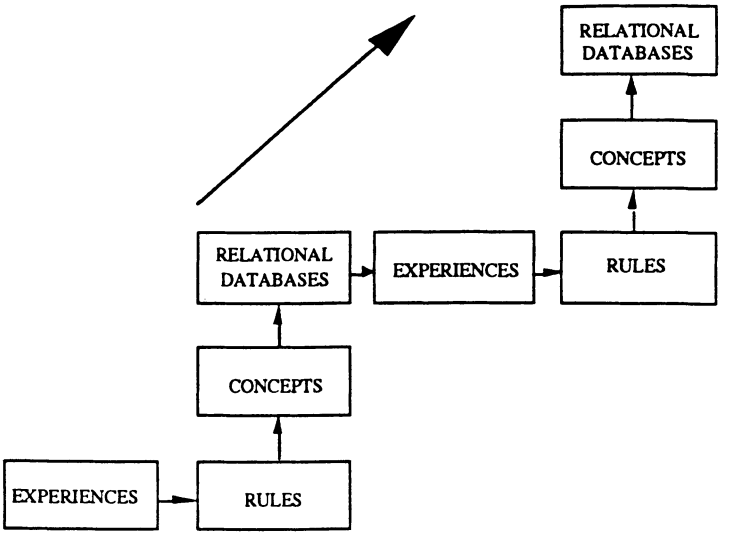


Figure 4. Evolution of Acquired Knowledge

2.4 Focusing Attention on the Subset of Admissible Experiences. This procedure is applied to the complete database of experiences. It outputs the set of admissible experiences. This is a procedure of *focusing attention.* At this stage, we narrow down the bulk of available data using the most general similarity feature, their "goodness" or "badness."

There are different options in selecting experiences suitable for generalization:

1. Select good/bad experiences which are close to each other in their value of goodness:
   a) experiences with the value of goodness higher/lower than a certain threshold.
   b) experiences which are a part of a particular top/bottom fraction of best/worst experiences (percentage threshold)
   c) experiences which form a group of n best/worst of them (quantity threshold).
2. Select good/bad experiences that are close to the current situation:
   a) experiences which are closer than a certain threshold to the current situation
   b) experiences which are the part of some top/bottom fraction of closer experiences
   c) experiences which form a group of n closest ones.
3. Intermediate strategies are possible. For example, if option 1 is applied and

doesn't find any rules with recommendations about the current situation, then option 2 can be applied. Option 2 can be used with different thresholds. If a large value of a threshold of closeness is chosen and no rules are found, then, for the current situation smaller threshold should be made. If no rule has been found after a few iterations with option 2, then, the learning algorithm should collect more experiences about the current situation.

When the database of experiences is large, the percentage threshold may choose too many experiences. A reasonable approach seems to be to select the "n" best experiences. In this case, we avoid setting a minimum value of goodness while retaining the ability to set the maximum amount of computational burden for the learning algorithm.

At the stage of combinatorial enhancement, focusing attention is applied by using some selected experiences to create enhanced situations and other to create enhanced actions. In this paper only addition and subtraction were mentioned for constructing enhanced situations and actions. Other operations can be used and they will generate larger sets of rules.

2.5 Formation of Similarity Clusters. In this subsection, we describe a process of grouping previously chosen good/bad experiences into clusters of similarity.
The procedure takes as an input the sets of experiences and enhanced experiences created in the previous steps and finds inner clusters among them. As shown in Figure 5, it inputs a set of groups of selected experiences and outputs clusters.

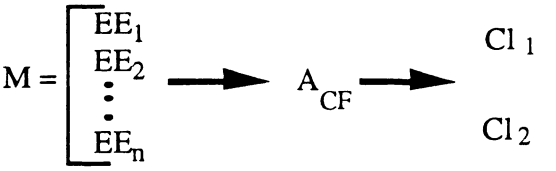


Figure 5. The Class Forming Algorithm

Clustering allows for reduction of computational complexity because of the following reasons:

- The system cannot store all particular experiences; creation of groups allows for compression of data.
- Many of experiences are ambivalent; the system needs only those experiences that can imply a set of actions or restrictions

The following requirements should be satisfied for the set of experiences used by clustering:

1. There are no repeated experiences in any of the output clusters

2. Every experience at the inputs should be included to one of the output clusters, i.e. the cluster forming algorithm does not eliminate any experience

3. No new experience are generated by the clustering algorithm

The comprehensive arsenal of theoretical tools for generating clusters can be found in [11]. We have explored two simple approaches for formation of classes: the "jump" approach and the closest neighbor approach. We would expect that the proper algorithm of clustering should be chosen taken in account all multiplicity of factors presented in the description of the environment within which the learning system is functioning.

The "jump" approach is similar to coordinate-by-coordinate approach introduced [12]. Instead of "sparseness," we use the term "density of good experiences."

Separating clusters by using the jump threshold leads to a large percentage of meaningless recommendations if clustering is performed on all experiences. However, in the initial set of experiences, only the good (or bad) ones are represented. So, the reason that we find jumps is because of inner cause-effect correlations. Only certain actions should be applied from the whole repertoire of possible actions. These actions can only be applied in certain clusters of situations to give acceptable (or unacceptable) values of goodness.

This will cause the algorithm to find rules of action only for unequivocally advantageous or totally unacceptable outcomes which should generate positive and/or negative rules. If no applicable rules can be found in a certain situation, then a new goal will be declared. This assigns a situation for which a good rule was found as a subgoal. Under this subgoal, a rule will be found or again a new subgoal will be declared.

The jump threshold is a measure of density of good experiences in the domain of a particular variable, because it separates the classes by means of finding domains of space that separate places where the good examples are situated densely. Most of the clusters separated by a jump become parts of meaningful rule hypotheses.

Another approach to clustering is merging based on a minimum distance or "the closest neighbor" method. Its strategy is to merge the most similar pairs of experiences at each step. The criterion used to select the closest experiences is similar to the idea applied in the stepwise optimal hierarchical methods. This makes the smallest possible stepwise increase in the sum of squared errors which is also a very common algorithm for finding clusters.

2.6 Searching for Valid Hypotheses among Clusters. The reason behind the creation of clusters is our belief that each of these clusters is a candidate to become a rule of a different behavior.

Also, we assume that there exists a class of enhanced representation actions A_{Enh} which has already been applied to a class of enhanced representation situations S_{Enh} and which produces values of goodness in the interval from J_{min} to J_{max}. If we have a situation belonging to a class S_{Enh}, then an action can be determined within class A_{Enh} which will provide goodness J_{max}. Then, each class of enhanced experiences becomes a hypothesis.

The hypotheses are then stored in the database of hypotheses as a tree where each hypothesis is related to its "parent" by the goal. If no hypothesis was found for a certain situation, then the situation in the hypothesis with closest situation to ours becomes a subgoal. This is one more source for the emerging hierarchies of acquired knowledge.

Figure 6 shows a parent hypothesis at the top, and a child hypothesis takes its goal from the parent's situation at the bottom. Since the child hypothesis has a new goal (which is the parent's situation) it has a new measure of goodness.

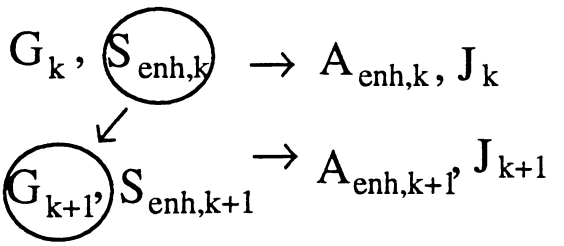


Figure 6. Parent and child hypothesis

It is not possible to find rules for all situations since only good experiences are selected to the classification procedure. Other parts of the space will use as a goal the situations with existing "direct" rule of action toward the original goal.

Two cases of hypotheses formation are to be considered: based upon situations (case A), and based upon actions (case B).

Case A. Suppose that two or more hypotheses in the database of hypotheses have the following characteristics:

1. They have the same goal.
2. They have the same enhanced action set.
3. They have different enhanced situation sets.

Then the following options exist:

1. Their situations do not intersect. In this case there are options:

a) leave them as separate hypotheses

b) create a joint hypothesis $\{S_{Enh1}, S_{Enh2}\} \rightarrow \{A_{Enh}\}$

2. They intersect, which again gives two options:

a) create a new joint hypothesis with situation $\{S_{Enh1}, S_{Enh2}\}$

b) create a new joint hypothesis with situation $\{S_{Enh.new}\}$.

Case B. Suppose that we have two or more hypotheses in the database of hypotheses that have the following characteristics:

1. They have the same goal.
2. They have the same enhanced situation.
3. They have the different enhanced action.

In this case it is necessary to merge their actions which produces two cases:

a) if their actions do not intersect then they will be left as separate hypotheses.

b) if they do intersect, a new joint hypothesis with action $\{A_{Enh.new}\}$ should be created.

3. Multiresolutional Search for a Motion Trajectory in the State Space: Top-Down Refinement

The second process characteristic for the learning automaton is looking for the preferable behavior [**4, 14-18**]. (The purpose of learning is to enable the subsequent process of planning.) This section discusses the process of deliberative planning. Conventional automata are only capable of reactive decisions; they are not capable of "look-ahead" decision making processes which are typical for deliberative planning. Unlike learning, which develops bottom-up and works via *generalization* (or coarsening), the process of planning develops top-down and works via *instantiation* (or refinement.) The process of planning is determined as choosing the desirable behavior by anticipating admissible alternatives among possible behaviors and selecting the best of them by comparing tentative trajectories in the state space (finding the *planned trajectory*, or PT.)

From Section 2, we conclude that all experiences acquired and hypotheses generated contain some knowledge of some reactive rules. For example, "if it is necessary to get to S_2 from S_1, apply A_{12}." This rule reacts by evoking A_{12} to the need of getting into S_2 from S_1 at i level of resolution. As this rule is applied top-down, it turns out that the space between $S_{1,i-1}$ and $S_{2,i-1}$ contains more nodes $S_{j,i-1}$, where j=1,2,3... are nodes of i-1 level of resolution. The process of selecting any string from them requires deliberation. Thus, multistep search procedures were recommended in [**12**].

PT is a trajectory which satisfies the specifications (the latter are determined by the

Goal). Trajectory is a string of adjacent admissible elementary subdomains (or *tessellata*, tiles of the discretized space) denoted as $\{\Omega_{ij}\}$ where i is the level of resolution and j is the number of the tessellatum in the string $\{\Omega_{ij}\}$. If necessary, PT can be also represented as a sequence of the subdomain indices $T_i\{j(\mu)\}$ where μ- is a number of elementary subdomains in a string $\{\mu=1,2,...,z\}$.

In this paper, we address only a well-posed problem (wpp) of planning. The proper set of wpp requires assigning: a) the initial point (SP_{i-1}) and the final point (FP_{i-1}) of the trajectory which should be determined at the higher resolution within the tessellatum of the resolution under consideration, and b) the feasible trajectory determined at the lower level of resolution.

PT is called a *feasible trajectory* if it has an initial point $SP_{i-1} \in \Omega_{i,1}$, a final point $FP_{i-1} \in \Omega_{i,z}$, and all tiles of the string are contained in the feasible trajectory at the lower level of resolution, or i+1 level.

Thus, a feasible trajectory for the level i is always represented as a string of tiles for the i+1 level of resolution and is a subspace of the i level of resolution. Clearly, the needs in and the algorithms of finding a feasible trajectory are tools of focusing attention. We determine the subspace in which the wpp of planning should be resolved, and the *optimum trajectory* should be found. Indeed, the feasible trajectory determined at the i+1 level of resolution becomes an "envelope," a bound domain of space at the i level of resolution.

The recursive definition of the feasible trajectory does not lead to any infinite incomputable computational procedure. As we extend our search for the feasible trajectory to the consecutively lower and lower levels of resolution, we eventually arrive at a level where both the initial point SP_i and the final point FP_i belong to the same tessellatum of the lower level of resolution FP_{i-1} SP_i, $FP_i \in \Omega_{i+1}$ which is the initial space for a feasible trajectory for the i level.

The sequence of feasible trajectories can be considered a nested system of spaces for multiresolutional search. To increase the reliability of searching for optimal trajectory, we increase the envelope of search by surrounding the feasible trajectory by one or more adjacent tiles along its boundary which will determine a particular "width" of the envelope. This results in a system of enhanced volumes $V_1>V_2>...>V_k>...>V_m$ which we will use later for searching (Figure 7).

The optimum trajectory at the i level of resolution is a *minimum-cost path* on the graph which is built upon all center-points of the tessellata at the i level of resolution. These tessellata are constructed within the envelope formed by the feasible

trajectory found for the i+1 level of resolution, as shown in Figure 7.

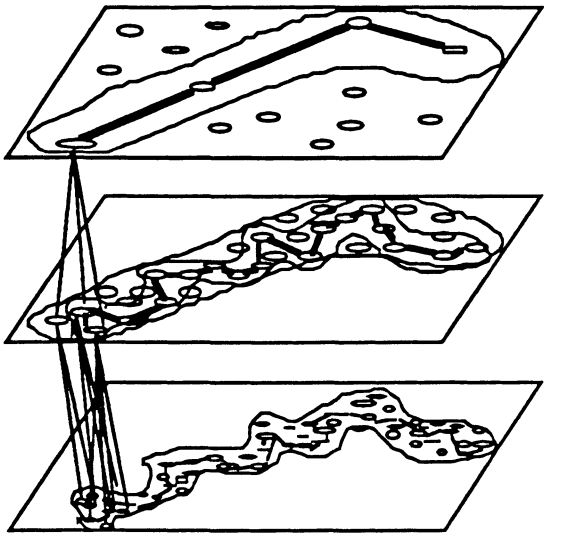


Figure 7. Illustration to the MS^3-algorithm.

The definitions for feasible and optimum trajectories imply the off-line method which has been introduced to find the best trajectory of motion to be followed by the control system. Search in the state space (S^3-search, see [**8, 12-18**]) is done by synthesizing the feasible trajectories for the i level of resolution and then building the alternatives of possible motion trajectories for the i-1 level within the envelope cost space. Domains of the state space Ω_k and their densities of points ρ_k at different levels are different so that the following inequalities hold

(5) $$\Omega_1 \supset \Omega_2 \supset \ldots \supset \Omega_k \supset \ldots \supset \Omega_m;$$

(6) $$\rho_1 < \rho_2 < \ldots < \rho_\kappa < \ldots < \rho_m;$$

while

(7) $$V_1 > V_2 > \ldots > V_k > \ldots > V_m;$$

where V_1 is the total volume of space under consideration. The results of *contraction* are introduced in Section 4. They depend on the system of resolution levels we deal with.

Some particular volume of the state space V_k is designated for a subsequent search for a solution, Operation of contraction puts constraints on this volume and should be

properly justified. We need to reduce the probability that contraction eliminates some or all of the opportunities to find *the* optimum path trajectory. The following heuristic strategy of contraction is chosen. After the search at the lowest resolution level is performed, the optimum trajectory is surrounded by an envelope. It is a convex hull which has a width w determined by the context of the problem. Then, the random points generation at the next level of resolution is performed only within this envelope of search. This strategy is demonstrated to be acceptable in many practical cases. However, the problem of consistency of representation under the contraction heuristic has to be addressed in the future.

4. Learning as a Part of Multiresolutional Intelligent Processing of Information

In this section, we introduce an algorithm that merges deliberative planning with learning. Both are more kindred to each other than we could expect. They produce and use the same multiresolutional system of representation. Let Ω-state space in which the start and final points SP and FP are given. The path from SP to FP is to be be found with the final accuracy ρ. Let us consider $\Omega=\Omega_1$ and $\rho=\rho_m$. We will introduce three operators.

Operator of Learning l (Ω,ρ) constructs the system of representation $(\Re)$ via creation of generalized maps $M = \Re\ (\Omega,\rho)$ at each level of resolution and obtained by generalization of the higher resolution map[1], ρ is the level of resolution of this map determined by the density of the search-graph.

There was no concept of "map" in the general description of the algorithm of l (Section 2). However, at the highest resolution level many high-resolutional tessellata of the space have a full representation. If representation of the tessellata are stored together with information about their adjacency, connections, and relations, then a set of dispersed data is being transformed into a consistent map. As the bottom-up process of generalization develops, the maps of the lower level of resolution emerge.

Even if the learning automaton was not given any idea of "self," this idea will be developed automatically by the algorithm of learning. At the lower resolution level, the deliberation of strings is expected, combined from cells of the higher level of resolution. Concatenation of these strings is possible only via concept of the "current state." It is possible to demonstrate that this concept will evolve into the concept of "self." Then, learning automaton will be able to put itself in a map.

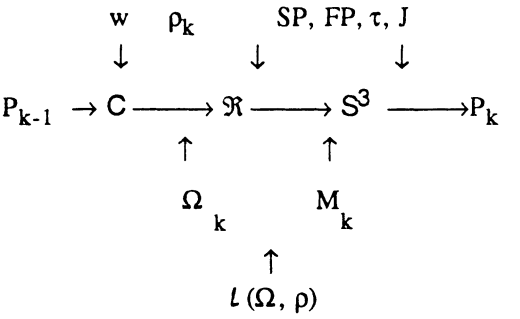
[1] The map of the highest level of resolution is obtained from the system of sensors transducing the signals of physical reality into some system of signs.

Operator of contraction $C:(P, w)\rightarrow\Omega = C(P,w)$ (where w is the parameter of the envelope, the "width" of the envelope) performs the procedure of focusing attention at each level of resolution (thus creating a system of volumes $V_1>V_2>...>V_k>...>V_m$). At the lowest level of resolution, focusing attention is determined by the total scope which is a part of assignment. At all levels below, the state space search should be formed for the consecutive search.

Operator of state space search S^3: $(V, SP, FP,\tau, J)\rightarrow P$ combines together in a form of a continuous string the tessellata within the envelope of search so that they form a continuous trajectory $P=S^3(V, SP, FP, \tau, J)$ from SP to FP. J is the cost (value of goodness) of behavior which should be minimized as a result of search S^3, V is the envelope of search, and τ is the value of time resolution at a particular level, or the interval of indistinguishability of time.

The structure for a class of multiresolutional algorithms of joint learning and behavior generation can be represented as a diagram 8.

(8)



We can see that the algorithms of this structure create a system of multiresolutional knowledge representation (bottom up) and a system of planning/control descriptions of the behavior to be conducted so that the system function successfully.

5. Evolution of Automata Equipped by Subsystems for Multiresolutional Unsupervised Learning and Behavior Generation

Figure 8 shows a detailed, enhanced version of the elementary loop of functioning [2,3]. This is a symbolic representation of the processing during various learning activities typical for a system equipped by modules unsupervised learning (UL) and behavior generation (BG). This remains the same in all goal-oriented cases of automata equipped with ULBG modules. Perception allows for recording the set of recent

experiences in a symbolic form. By grouping the experiences, the classes of similarity are discovered. This induce hypotheses explaining the similarities, or instigate new experiences belonging to the same class of similarity. Within the semiotic paradigm, the loop of ULBG can be called "a loop of semiosis" [2].

The system is presumed to function under an externally assigned Goal. The initial set of experiences (which might be obtained by random actions) are generalized into hypotheses. The hypotheses enter the subsystem of Behavior Generation as a substitute for the rules. The decision for an action is made; the action is performed; changes in the world occur; the transducers (sensors) transform them into a form that can be used by Perception. The long and complicated process of moving from signs to meaning starts again. Now, the enhanced set of experiences brings about another hypotheses which can confirm or refute the tested ones. This is when the *symbol grounding* happens.

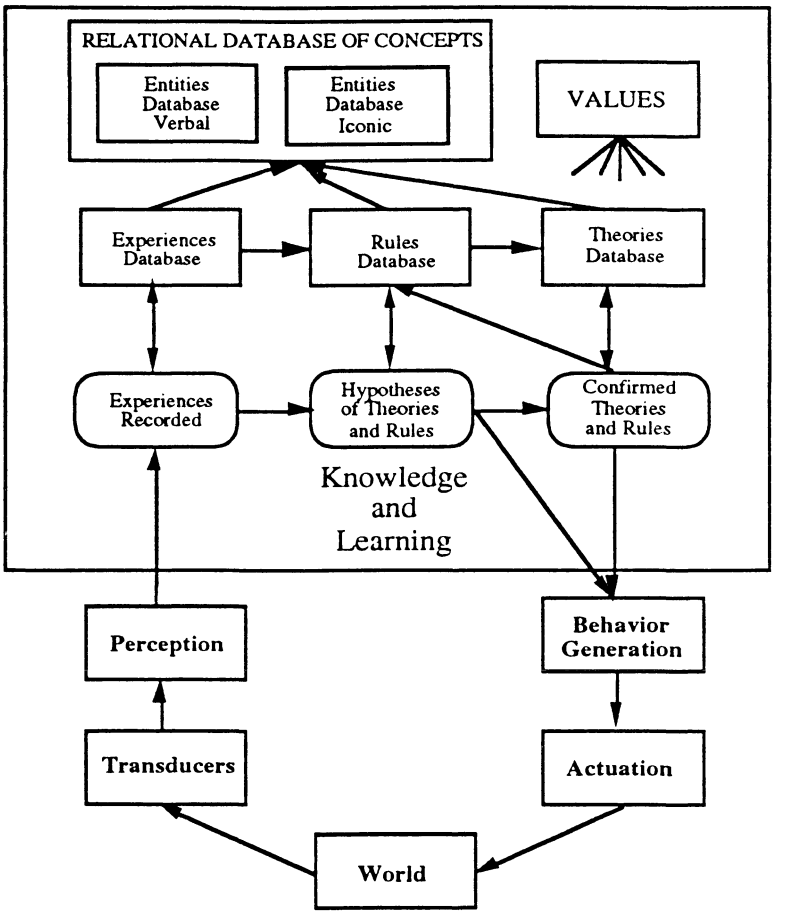


Figure 8. Functioning of Learning Automaton: Six Box Diagram

After multiple tests, the hypotheses can cross the threshold of "trustworthiness," and a new rule is created. A rule (or a set of rules) within a context is considered to be "a theory." At each development step, the unit under consideration undergoes a comparison with other kindred units confined in corresponding databases (of Experiences, of Rules, and of Theories.) Then, the symbols tentatively assigned to some "unities," "entities," or "concepts" enter their place within the database of concepts (which is a relational network of symbols.)

Rules (or the hypotheses which will become rules) are formed when experiences cluster together unified by their similarity. For the prior state S_1 the applied action A_{12} leads to the emergence of the new state S_2; the value of reward J_{12} is the result. After gathering a sufficient number of the experiences and proper generalizing them the rules of the following form can be constructed: IF the value J is desired upon achievement of a the goal-state S_G from the present state S_1, THEN the action A_{1G} should be applied.

An interesting and unique feature of generating rules follows. Each component of a rule is a generalized component of experience. This means that to obtain a component of a rule, several similar components of experiences should be grouped together into a class, a cluster. This requires applying a set of procedures using the triplet of grouping, focusing attention, and combinatorial search. The label attached to this cluster signifies the process and the result of generalization. The premises behind the process of generalization could be different. But, the result will be always the same: creation of the new object for the lower level of resolution. For example, let $\mathcal{G}_i$ symbolizes the phenomenon of generalization upon i similar experiences (i=1, 2, ..., n), then

(9) [The zone of states "S" with $J_1<J<J_2$] $\rightarrow \mathcal{G}_i\{S_i, J_1<J_i<J_2\}$,

(10) [Action "A" to be applied to achieve a desired zone] $\rightarrow$ $\mathcal{G}_i\{S_i, A_{i,i+1}\}$.

Only the desired state is not subject to generalization. It is always individual, pertaining to a concrete system and problem.

6. Conclusions: The Issues of Further Research

Theoretical analysis presented in this paper has been confirmed by the experimental results [**1**, **7**, **8**]. "Baby-Robot" was able to learn how to reach the arbitrary situated goal only after implementing the algorithm of generalization with combinatorial enhancement. Before this algorithm was implemented, Baby-Robot was

able to learn how to reach the particular located goal. If the location of the goal was changed, the successful learning process for the previous goal could not help to find a new one. Generalization n with combinatorial enhancement has enabled the robot to make the discovery, and initiate the process of hierarchical learning. Other positive results are recorded in [17,18].

1. In this paper, we introduce and analyze an algorithm of multiresolutional unsupervised learning with inductive generalization and a search for hidden implications. This algorithm is applied recursively to its own results at the output. Therefore, it builds up a multiresolutional structure of results. This type of unsupervised learning (UL) is demonstrated to be a mechanism of development of an evolving multiresolutional system of representation. It enables the system of behavior generation (BG) also to evolve. Together, (ULBG) they provide for an evolution of the automata equipped with such systems. The automaton equipped by ULBG becomes a multiresolutional automaton, and its levels of resolution can change as the evolution of knowledge and behavior proceeds. This evolution can be illustrated by Figure 9.

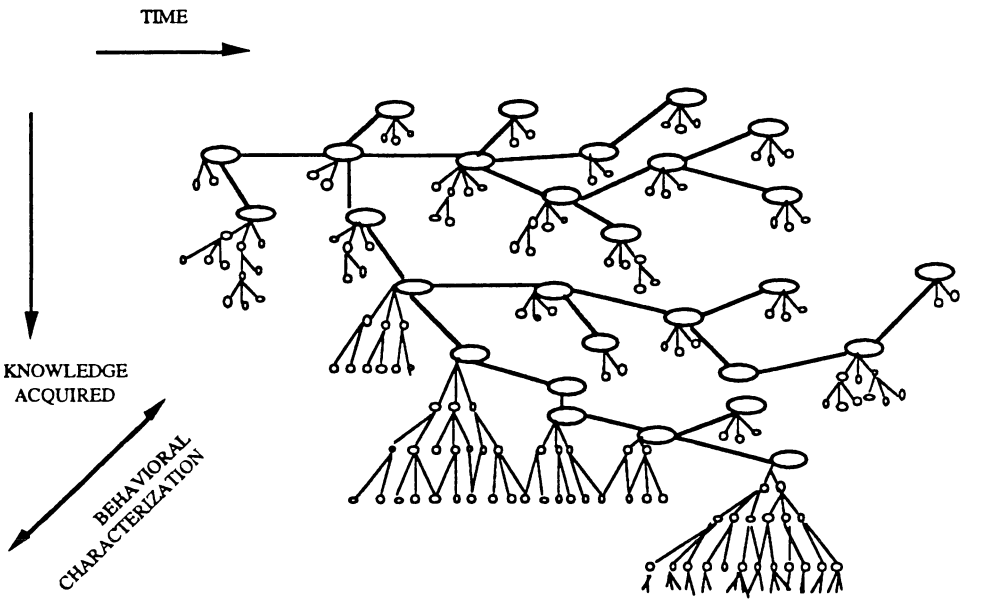


Figure 9. Evolution of Knowledge and Behavior of the Automata with ULBG

2. From Figure 9, one can see the behavior evolves. This producing different plans and motion trajectories as shown as a horizontally developing tree. At the same time, its knowledge evolves as shown in the vertical hierarchical structures.

This process seems to be even more important to analyze of the evolution of living creatures. We believe that the automaton equipped by ULBG allows for analysis of the processes of evolution of these systems (automata with ULBG) as *species*. It is possible to equip the automaton by the system of reproduction. It would be possible to analyze how the process of *knowledge evolution* is affected by different mechanisms of reproduction.

3. This line of research takes advantage of the uniqueness of automata with ULBG among other known systems of automata with learning. The mechanism of unsupervised learning allows for the ultimate freedom in the way the learning process organizes the acquired knowledge. It is possible to anticipate that as the knowledge base evolves, the knowledge becomes utterly diversified. Rules concerning the external world will emerge, and the rules concerning processes of inner knowledge organization and procedures of processing will follow. Simple mental experiments can confirm that the system which starts with perceiving the world as a set of values $\{v_{ij}\}$, i.e. having an "ego-focused" representation will learn how to develop maps in externally-fixed coordinates that will allow for putting on the map the system itself (which might be interpreted as emergence of "consciousness" in some applications).

4. The automaton with ULBG can be used to analyze all stages of learning including the "early learning" stage. Certainly, some initial knowledge ("bootstrap knowledge") is presumed. This bootstrap organization of knowledge can strongly affect the subsequent processes of knowledge evolution. On the other hand, the learning system is presumed to be free of a building up of all subsequent knowledge organization. How it will organize the knowledge acquired and why?— this is the research issue for the are a of automata with ULBG.

5. The processes of knowledge acquisition are affected by the knowledge stored. They start creating some bias in the subsequent knowledge acquisition since the results of automaton functioning will be induced by the knowledge previously stored. So, if the results of functioning were "good" or led to a "better" behavior, the system might assume that its goodness is due to the knowledge used. It might happen that the experiences the system acquires are limited by its predisposition. This is another research topic of interest in the ULBG area.

6. Since the system has been developed to demonstrate some particular behavior, the evaluation of this behavior should be the ultimate measure for the process and results of knowledge organization as well as the processes and results of the ways knowledge is acquired from the external world and knowledge used for behavior generation. This

determines an interesting interconnection between further developing the hierarchy of functions for evaluation of goodness which precipitate at different levels of resolution. The interconnection between learning, behavior, and developing of the "system of values" in the automata with ULBG seems to be an important research issue.

7. In concert, all these three processes: acquisition, organization and use affect the overall system functioning. Therefore, other systems of learning can modify and alter the system of ULBG.At the present time the following mechanisms of knowledge acquisition are known from the literature (other than UL):

a). Learning by transfer. In this case all knowledge which subsequently is required for behavior generation is transferred from another source where it was stored and organized in advance based upon existing design decisions and experiences of functioning. This method of knowledge acquisition presumes that the structure of the system of interest and its functioning in required circumstances are previously known, and knowledge is organized so that this structure be properly supported.

b). Learning by examples. In this case, we presume a "Teacher" which has substantial knowledge about my cases of possible functioning, stores knowledge of previous experiences and spells out a set of possible scenarios in which the functioning of the system is expected. Undoubtedly, a set of tests can be developed in which a behavior of system is entertained and after each case of behavior the system receives the teacher's evaluation whether it was good, and how good it was. These tests teaches exercises for which the solution is known. Interaction of ULBG system with other systems of learning should be a separate research issue.

The system is to be taught the responses to the test by demonstrating a particular behavior. At the end of the test, it is informed by the teacher whether this behavior was right or wrong. In more complex schemes, it can be informed of how good it was and why. Unlike in the mechanism 1, the mechanism 2 does not interfere with how the learning system organizes the required knowledge. However, the teacher's interference into the process of knowledge acquisition can be deep enough. The mechanism 2 does not say anything about the way knowledge should be organized within the learning system.

8. Learning and Behavior Generation produce structural and behavioral hierarchies. It is possible to state that using ULBG reduces computational complexity by increasing the structural complexity.

7. References

1. A. Meystel, *Baby-robot: On the analysis of cognitive controllers for robotics*, Proc. IEEE Int'l Conf. on Man & Cybernetics, Tuscon, AZ, Nov. 11-15 (1985), 327-222

2. _____ , *Intelligent Systems: A Semiotic Perspective*, International Journal of Intelligent Control and Systems, Vol.1, No.1, (1996), 31-57
3. J. Albus, A. Meystel, *A Reference Model Architecture for Design and Implementation of Semiotic Control in Large and Complex Systems*, Proc. of IEEE ISIC Workshop, Architectures for Semiotic Modeling and Analysis in Large Complex Systems, Monterey, CA (1995), 33-45
4. A. Meystel, *Nested Hierarchical Control*, in Eds. P. Antsaklis, K. Passino, An Introduction to Intelligent and Autonomous Control, Kluwer Academic Publisher, Boston, MA 1992
5. K. Rajaraman, P.S. Sastry, *Finite Time Analysis of the Pursuit Algorithm for Learning Automata*, IEEE Transactions on Systems, Man & Cybernetics, Part B: Cybernetics, Vol.26, No.4, (1996), 590-598.
6. J. Albus, *Outline for a Theory of Intelligence*, IEEE Transactions on Systems, Man and Cybernetics,Vol. 21, No. 3, (1991), 473-509
7. J. Albus, A. Lacaze, A. Meystel, *Autonomous Learning Via Nested Clustering*, Proc. of the 34th IEEE Conference on Decision and Control, New Orleans LA, (1995), 3034-3039.
8. _____ , *Theory and Experimental Analysis of Cognitive Processes in Early Learning*, Proc. of the 1995 IEEE International Conference on Systems, Man and Cybernetics, vol. 4, Vancouver, BC, Canada, (1995), 4404-4409
9. D.B. Fogel, *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence*, IEEE Press, New York, 1995
10. A. Meystel, M. Thomas *Computer aided conceptual design in robotics*, Proc. IEEE Int'l Conf. in Robotics, Atlanta, GA, March 13-15 (1984), 220-229
11. B. Mirkin, *Mathematical Classification and Clustering*, Kluwer Academic, Dordrecht, 1996
12. R.S. Michalski, *A Theory and Methodology of Inductive Learning*, Artificial Intelligence, Vol. 20, No.2, Elsevier, Amsterdam, 1989.
13. L. Goldfarb, J. Abela, V.C. Bhavsar, V.H. Kamat, *Can a Vector Space Based Learning Model Discover Inductive Class Generalization* in a Symbol Environment, Pattern Recognition Letters, V. 16, (1995), 719-726
14. A. Meystel, *Planning in a hierarchical nested controller for autonomous robots*, Proc. IEEE 25th Conf. on Decision and Control, Athens, Greece, 1986
15. _____ , *Planning in a Hierarchical Nested Control System*, in Eds. W. Wolfe, N. Marquina, Mobile Robots, Proc. of SPIE, Vol. 727, Cambridge, MA (1986), 42-76
16. Saridis G., *An Integrated Theory of Intelligent Machines by Expressing the Control Performance as Entropy*, Control: Theory and Advanced Technology, Vol.1, No. 2, MITA-Press, Tokyo, Japan, (1985), 125-138.
17. A. Meystel, S. Uzzaman, G. Landa, S. Wahi, R. Navathe, B. Cleveland, *State Space Search For An Optimal Trajectory*, Proc. of the IEEE Symposium on Intelligent Control, Vol. II, Philadelphia, PA, (1990)
18. Grevera G., Meystel A., *Searching for a path through Pasadena*, Proc. of the IEEE Symposium on Intelligent Control, Arlington, VA,1989

Intelligent Systems Division
National Institute of Standards and Technology,
Gaithersburg, MD 20899
On leave from Drexel University, Philadelphia PA 19104

E-mail address: meystel@cme.nist.gov

This page intentionally left blank

https://doi.org/10.1090/dimacs/037/25

Index

Selected Titles in This Series

(*Continued from the front of this publication*)

6 **Jacob E. Goodman, Richard Pollack, and William Steiger, Editors,** Discrete and Computational Geometry: Papers from the DIMACS Special Year

5 **Frank Hwang, Fred Roberts, and Clyde Monma, Editors,** Reliability of Computer and Communication Networks

4 **Peter Gritzmann and Bernd Sturmfels, Editors,** Applied Geometry and Discrete Mathematics, The Victor Klee Festschrift

3 **E. M. Clarke and R. P. Kurshan, Editors,** Computer-Aided Verification '90

2 **Joan Feigenbaum and Michael Merritt, Editors,** Distributed Computing and Cryptography

1 **William Cook and Paul D. Seymour, Editors,** Polyhedral Combinatorics

ISBN 0-8218-0762-5