Les Cahiers du GERAD ISSN: 0711–2440

Cluster Analysis and Mathematical Programming

Pierre Hansen
Brigitte Jaumard

G–97–10

March 1997

Cluster Analysis and Mathematical Programming

Pierre Hansen
GERAD and École des Hautes Études Commerciales
Montréal, Canada

Brigitte Jaumard
GERAD and École Polytechnique de Montréal
Canada

February, 1997

Les Cahiers du GERAD
G–97–10

Abstract

Given a set of entities, Cluster Analysis aims at finding subsets, called clusters, which are homogeneous and/or well separated. As many types of clustering and criteria for homogeneity or separation are of interest, this is a vast field. A survey is given from a mathematical programming viewpoint. Steps of a clustering study, types of clustering and criteria are discussed. Then algorithms for hierarchical, partitioning, sequential, and additive clustering are studied. Emphasis is on solution methods, i.e., dynamic programming, graph theoretical algorithms, branch-and-bound, cutting planes, column generation and heuristics.

Résumé

Étant donné un ensemble d'objets, la classification automatique a pour but de trouver des sous-ensembles, ou classes, homogènes et/ou bien séparées. Comme de nombreux types de classification et critères d'homogénéité et de séparation sont dignes d'intéret, ce domaine est varié. On en présente une revue, d'un point de vue de programmation mathématique. On discute les étapes d'une étude de classification, les types de classigication et les critères. On étudie ensuite les algorithmes de classification hiérarchique, de partitionnement, de classification séquentielle et additive. On insiste sur les méthodes de résolution, c'est-à-dire la programmation dynamique, les algorithmes de graphes, les procédures d'optimisation par séparation, la génération de colonnes et les heuristiques.

Acknoledgment: Corresponding author. Research supported by ONR grant N00014-95-1-0917, FCAR grant 95-ER-1048 and NSERC grants GP0105574 and GP0036426.

State-of-the-art survey to be presented at the XVIth Mathematical Programming Symposium, Lausanne August 25–29 1997, to appear in *Mathematical Programming, B.*

1 Introduction

Consider a set of entities together with observations or measurements describing them. Cluster Analysis deals with the problem of finding subsets of interest called *clusters*, within such a set. Usually, clusters are required to be *homogeneous* and/or *well separated*. Homogeneity means that entities within the same cluster should resemble one another and separation that entities in different clusters should differ one from the other [15]. This problem is old. It can be traced back to Aristotle and was already much studied by XVIIIth century naturalists such as Buffon, Cuvier and Linné. It is also ubiquitous, with applications in the natural sciences, psychology, medicine, engineering, economics, marketing and other fields. As a consequence, the cluster analysis literature is vast and heterogeneous (the yearly *Classification Literature Automated Search Service* lists many books and hundreds of papers on that topic in each issue). Cluster analysis algorithms draw upon statistics, mathematics and computer science. Closely related fields are pattern recognition, computer vision, computational geometry and subfields of operations research such as location theory and scheduling.

Given a cluster analysis problem, the following questions should be answered:

- what is the aim of the clustering — the question of *criterion* (or criteria);
- are we justified in pursuing that aim — the question of *axiomatics*;
- what constraints should be considered — the question of *choice of clustering type*;
- how difficult is it to perform the clustering — the question of *complexity*;
- how should the clustering be done — the question of *algorithm design*;
- is the clustering obtained meaningful — the question of *interpretation*.

While each of these questions has been studied, sometimes extensively, only some chapters of cluster analysis (mainly agglomerative hierarchical clustering) appear to be thoroughly explored, i.e., expressed as well developed mathematical theories. A fruitful way to address the questions listed above (except possibly the last one) is to adopt a mathematical programming viewpoint. While a few clustering problems were expressed as mathematical programs before, systematic use of that approach was only advocated about 25 years ago [114, 102]. The purpose of the present paper is to review the mathematical programming approach to cluster analysis since that time. No attempt will be made to be exhaustive. We will focus on the large class of methods which use dissimilarities. We hope, however, to give a fairly representative view of the main classes of clustering problems within that paradigm and of the most efficient tools to solve them. The paper is organized as follows. Ingredients of cluster

analysis are reviewed in the next section: steps of a cluster analysis study, types of clusterings and criteria. Section 3 is devoted to hierarchical clustering. Agglomerative and divisive algorithms are reviewed. Section 4 addresses partitioning problems, and is organized by solution technique. Six of them are considered: dynamic programming, graph theoretical algorithms, branch-and-bound, cutting planes, column generation and heuristics. Other less frequently used clustering paradigms are examined in Section 5: sequential clustering, additive clustering and representation of dissimilarities by trees. Brief conclusions are drawn in Section 6.

2 Ingredients of Cluster Analysis

2.1 Steps of a cluster analysis study

Most cluster analysis methods rely upon dissimilarities (or similarities, or proximities) between entities, i.e., numerical values either directly observed or, more often, computed from the data before clustering. A general scheme for dissimilarity-based clustering is the following:

(a) Sample. Select a sample $O = \{O_1, O_2, \ldots, O_N\}$ of N entities among which clusters are to be found.

(b) Data. Observe or measure p characteristics of the entities of O. This yields a $N \times p$ data matrix X.

(c) Dissimilarities. Compute from the matrix X a $N \times N$ matrix $D = (d_{k\ell})$ of dissimilarities between entities. Such dissimilarities (usually) satisfy the properties $d_{k\ell} \geq 0$, $d_{kk} = 0$, $d_{k\ell} = d_{\ell k}$ for $k, \ell = 1, 2, \ldots, N$. They need not satisfy the triangle inequality, i.e., be distances.

(d) Constraints. Choose the type of clustering desired (hierarchy of partitions, partition, ...). Specify also further constraints on the clusters, if any (maximum weight or cardinality, connectedness, ...).

(e) Criterion. Choose a criterion (or possibly two criteria) to express homogeneity and/or separation of the clusters in the clustering to be found.

(f) Algorithm. Choose or design an algorithm for the problem defined in (d), (e). Obtain or write the corresponding software.

(g) Computation. Apply the chosen algorithm to matrix $D = (d_{k\ell})$ thus obtaining clusters, and clusterings of the chosen type.

(h) Interpretation. Apply formal or informal tests to select the best clustering(s) among those obtained in (g). Describe clusters by their lists of entities and descriptive statistics. Proceed to a substantive interpretation of the results.

Steps (d) and (e) define a clustering problem as a mathematical program. Steps (a) to (c) and (h) correspond to a statistical viewpoint on clustering. They are in many ways delicate and discussed at length in the literature [111, 73, 48, 83]. We focus here on steps (d) to (g) which correspond to a mathematical programming viewpoint.

Several remarks are in order. First, dissimilarities may be computed from other sources than a matrix of measurements X, for instance when comparing biological sequences or partitions. Second, for some methods only the order of the dissimilarities matters. This information can be obtained by questions such as " are these two entities more similar than these two other ones". Third, cluster analysis is not the only way to study dissimilarities or distances between entities in the field of *data analysis*. Another much used technique is *principal component analysis* (e.g. [99]). Fourth, few assumptions are made on the clusters in the above scheme and they are usually in set-theoretic terms. In some circumstances, more knowledge is available. For instances, the set of entities may be associated with a mixture of distributions, the number and parameters of which are to be found (e.g. [96] chap. 3). Or yet clusters may correspond to given objects such as characters, to be recognized. This last case pertains to *pattern recognition*, a field close to but different from cluster analysis. Fifth, instead of computing dissimilarities, *direct clustering* may be performed on the matrix X. An early example is maximization of the *bond-energy* or sum for all cells of products of their values with the values of adjacent cells [92]. Heuristics are based on permuting rows and columns, and an exact solution is obtained by solving two associated traveling salesman problems [89]. Clusters found by direct clustering may be interpreted in conceptual terms. Recently, conceptual clustering has become a very active field of research (e.g. [34, 106]).

2.2 Types of clustering

Cluster analysis algorithms are designed to find various types of clusterings, e.g.,

(i) **Subset** C of O;

(ii) **Partition** $P_M = \{C_1, C_2, \ldots, C_M\}$ of O into M clusters;
(ii a) $C_j \neq \emptyset \quad j = 1, 2, \ldots, M$;
(ii b) $C_i \cap C_j = \emptyset \quad i, j = 1, 2, \ldots, M$ and $i \neq j$;
(ii c) $\bigcup_{i=1}^{M} C_j = O$;

(iii) **Packing** $Pa_M = \{C_1, C_2, \ldots, C_M\}$ of O with M clusters:
as (ii) but without (ii c);

(iv) **Covering** $Co_M = \{C_1, C_2, \ldots, C_M\}$ of O by M clusters:
as (ii) but without (ii b);

(v) **Hierarchy** $H = \{P_1, P_2, \ldots, P_q\}$ of $q \leq N$ partitions of O.
Set of partitions $P_1, P_2, \ldots, P_q$ of O such that $C_i \in P_k$, $C_j \in P_\ell$ and $k > \ell$ imply $C_i \subset C_j$ or $C_i \cap C_j = \emptyset$ for all $i, j \neq i$, $k, \ell = 1, 2, \ldots, N$.

By far the most used types of clustering are the partition and the complete hierarchy of partitions, i.e., that one containing N partitions. This last hierarchy can also be defined as a set of $2N - 1$ clusters which are pairwise disjoint or included one into the other. Recently, weakenings of hierarchies are also increasingly studied. They include *hierarchies of packings* [91], *weak hierarchies* [2] and *pyramids* [35]. Work has also been done on *fuzzy clustering*, in which entities have a degree of membership in one or several clusters [10].

In constrained clustering, additional requirements are imposed on the clusters. The most frequent are bounds on their cardinality, bounds on their weight, assuming entities to be weighted, or connectedness, assuming an adjacency matrix between entities is given.

2.3 Criteria

We first consider dissimilarity-based criteria used to express separation or homogeneity of a single cluster C_j. Separation of C_j can be measured by:
(i) the *split* $s(C_j)$ of C_j, or minimum dissimilarity between an entity of C_j and one outside C_j:

$$s(C_j) = \min_{k: O_k \in C_j,\ \ell: O_\ell \notin C_j} d_{k\ell};$$

(ii) the *cut* $c(C_j)$ of C_j, or sum of dissimilarities between entities of C_j and entities outside C_j:

$$c(C_j) = \sum_{k:O_k \in C_j} \sum_{\ell:O_\ell \notin C_j} d_{k\ell}$$

and one might also consider a *normalized cut*, which corrects the previous measure to eliminate the effect of the cluster's size by dividing $c(C_j)$ by $|C_j|\Big(N - |C_j|\Big)$.

Homogeneity of C_j can be measured by:
(i) the *diameter* $d(C_j)$ of C_j, or maximum dissimilarity between entities of C_j:

$$d(C_j) = \underset{k,\ell:O_k,O_\ell \in C_j}{\text{Max}} d_{k\ell};$$

(ii) the *radius* $r(C_j)$ of C_j or minimum for all entities O_k of C_j of the maximum dissimilarity between O_k and another entity of C_j:

$$r(C_j) = \underset{k:O_k \in C_j}{\text{Min}} \underset{\ell:O_\ell \in C_j}{\text{Max}} d_{k\ell};$$

(iii) the *star* $st(C_j)$ of C_j or minimum for all entities O_k of C_j of the sum of dissimilarities between O_k and the other entities of C_j:

$$st(C_j) = \underset{k:O_k \in C_j}{\text{Min}} \sum_{\ell:O_\ell \in C_j} d_{k\ell};$$

(iv) the *clique* $c\ell(C_j)$ of C_j or sum of dissimilarities between entities of C_j;

$$c\ell(C_j) = \sum_{k,\ell:O_k,O_\ell \in C_j} d_{k\ell};$$

and one might also consider a *normalized star* and a *normalized clique* defined as $st(C_j)$ divided by $|C_j| - 1$ and $c\ell(C_j)$ divided by $|C_j|\Big(|C_j| - 1\Big)$ respectively.

If the entities O_j are points x of a p-dimensional Euclidean space, further concepts are useful. Homogeneity of C_j is then measured by reference to a center of C_j which is no more a point of C_j, as in the definitions of $r(C_j)$ and $st(C_j)$. One can then use (i) the *sum-of-squares* $ss(C_j)$ of C_j or sum of squared Euclidean distances between entities of C_j and its centroid $\overline{x}$:

$$ss(C_j) = \sum_{k:O_k \in C_j} \Big(||x_k - \overline{x}||_2\Big)^2$$

where $||\cdot||_2$ denotes Euclidean distance and

$$\overline{x} = \frac{1}{|C_j|} \sum_{k:O_k \in C_j} x_k;$$

(ii) the *variance* $v(C_j)$ of C_j defined as $ss(C_j)$ divided by $|C_j|$;
(iii) the *continuous radius* $cr(C_j)$ of C_j defined by

$$cr(C_j) = \underset{x \in \mathbb{R}^p}{\text{Min}} \underset{k:O_k \in C_j}{\text{Max}} ||x_k - x||_2;$$

(iv) the *continuous star* $cst(C_j)$ of C_j defined by

$$cst(C_j) = \underset{x \in \mathbb{R}^p}{\text{Min}} \sum_{k:O_k \in C_j} ||x_k - x||_2.$$

Next, consider partitions P of O into M clusters. The concepts defined above yield, in a straightforward way, two families of criteria, to be maximized for separation and minimized for homogeneity. They correspond to focusing on the worst cluster or considering all clusters (or average values) respectively. So the *split* $s(P_M)$ of partition P_M is the smallest split of its clusters, the *diameter* $d(P_M)$ of P_M is the largest diameter of its clusters, and so on. The *average split* $av(P_M)$ of P_M is the sum of splits of its clusters divided by M, the *average diameter* $ad(P_M)$ of P_M is the sum of diameters of its clusters divided by M, and the like.

Similar definitions can be given for packings, coverings and hierarchies (viewed as sets of $2N - 1$ clusters).

Again, several remarks are in order. First, not all criteria are independent. For instance, minimizing average clique is equivalent to maximizing average cut. Second, a few criteria express both homogeneity and separation. This is the case for minimizing the within-clusters sum-of-squares, a criterion of homogeneity, which is equivalent to maximizing the between-clusters sum of squares, a criterion of separation. Third, values of $s(P_M)$, $r(P_M)$ and $d(P_M)$ are equal to a single dissimilarity value. Hence, there are few potential values. Moreover, the optimal partitions are not modified by a monotone transformation of the dissimilarities. Fourth, criteria such as $r(C_j)$, $st(C_j)$, $ss(C_j)$ and $v(C_j)$ make use of a cluster center. This center may be usefully considered as representative of the cluster in some applications. Fifth, criteria defined for partitions can be used in several ways: they can be optimized globally (exactly or approximately) in partitioning or locally in hierarchical clustering, where changes from a

partition to the next are subject to constraints. Sixth, asymmetric dissimilarities may be reduced to symmetric dissimilarities, e.g. by taking minimum or maximum values associated to opposite directions for each pair of entities. Alternately, definitions given above may be adapted [76].

Criteria used in additive clustering differ form those described here, and will be examined in Section 5.

3 Hierarchical Clustering

3.1 Agglomerative hierarchical clustering algorithms

Agglomerative hierarchical clustering algorithms are among the oldest and still most used methods of cluster analysis [23, 49]. They proceed from an initial partition in N single-entity clusters by successive mergings of clusters until all entities belong to the same cluster. Thus, they fit into the following scheme:

Initialization
$P_N = \{C_1, C_2, \ldots, C_N\}$;
$C_j = \{O_j\} \quad j = 1, 2, \ldots, N$;
$k = 1$;
Current step:
While $N - k > 1$ **do**
select $C_i, C_j \in P_{N-k+1}$ following a local criterion;
$C_{N+k} = C_i \cup C_j$;
$P_{N-k} = \Big(P_{N-k+1} \cup \{C_{N+k}\}\Big) \setminus \{C_i, C_j\}$;
$k = k + 1$
EndWhile

By a local criterion, we mean a criterion which uses only the information given in D and the current partition. Thus the algorithm uses no memory about how this partition was reached or look-ahead feature about other partitions than the next one.

Many local criteria have been considered. They correspond to criteria for the partitions obtained, sometimes defined in an implicit way. This is the case for the *single-linkage* algorithm, which merges at each step the two clusters for which the smallest inter-cluster dissimilarity is minimum. Indeed, a well-known graph theoretic

result of [105] can be reformulated as follows. Let $G = (V, E)$ denote a complete graph, with vertices v_k associated with entities O_k, for $k = 1, 2, \ldots, N$ and edges $\{v_k, v_\ell\}$ weighted by the dissimilarities $d_{k\ell}$. Let MST denote a minimum spanning tree of G.

Proposition 1 *[105] The values of the split for all subsets of entities of O, and hence for all partitions of O, belong to the set of dissimilarity values associated with the edges of MST.*

Corollary 1 *[28] The single-linkage algorithm provides maximum split partitions at all levels of the hierarchy.*

For other criteria, the partitions obtained after several steps of an agglomerative algorithm are not necessary optimal. For instance, the *complete-linkage* algorithm merges at each step the two clusters for which the resulting cluster, as well as the resulting partition, has smallest diameter. After two steps or more this partition may not have minimum diameter. An algorithm to find minimum diameter partitions is discussed in the next section.

An interesting updating scheme for dissimilarities in agglomerative hierarchical clustering has been proposed in [87] and extended in [79, 80]. A parametric formula gives new dissimilarity values between cluster C_k and C_i, C_j when these last two are merged:

$$d_{k,i\cup j} = \alpha_i d_{ik} + \alpha_j d_{jk} + \beta d_{ij} + \delta |d_{ik} - d_{jk}|.$$

Values of the parameters, a few examples of which are given in Table 1, correspond to single-linkage, complete-linkage and other methods. Clusters to be merged at each iteration are those corresponding to the smallest updated dissimilarity. Using heaps, an $O(N^2 \log N)$ uniform implementation of agglomerative hierarchical clustering is obtained [26].

Better results can be derived in a few cases: finding the MST of G, ranking its edges by non-decreasing values and merging entities at endpoints of successive edges yields a $\theta(N^2)$ implementation of the single-linkage algorithm [50]. At each iteration, clusters correspond to connected components of a graph with the same vertex set as G and as edges those of MST considered. A $\theta(N^2)$ algorithm based on similar principles has also been obtained [112] for clustering with asymmetric dissimilarities and strongly connected components as clusters.

Table 1: Coefficients in updating formula for agglomerative hierarchical clustering

Method	α_i	α_j	β	δ
Single linkage	$1/2$	$1/2$	0	$-1/2$
Complete linkage	$1/2$	$1/2$	0	$1/2$
Average linkage	$\frac{\lvert C_i\rvert}{\lvert C_i\rvert+\lvert C_j\rvert}$	$\frac{\lvert C_j\rvert}{\lvert C_i\rvert+\lvert C_j\rvert}$	0	0
Centroid	$\frac{\lvert C_i\rvert}{\lvert C_i\rvert+\lvert C_j\rvert}$	$\frac{\lvert C_j\rvert}{\lvert C_i\rvert+\lvert C_j\rvert}$	$\frac{-\lvert C_i\rvert\lvert C_j\rvert}{(\lvert C_i\rvert+\lvert C_j\rvert)^2}$	0
Ward's method	$\frac{\lvert C_i\rvert+\lvert C_k\rvert}{\lvert C_i\rvert+\lvert C_j\rvert+\lvert C_k\rvert}$	$\frac{\lvert C_j\rvert+\lvert C_k\rvert}{\lvert C_i\rvert+\lvert C_j\rvert+\lvert C_k\rvert}$	$\frac{-\lvert C_k\rvert}{\lvert C_i\rvert+\lvert C_j\rvert+\lvert C_k\rvert}$	0

The following *reducibility property* has been studied in [13]:

$$d(C_i, C_j) \leq \min\Big\{d(C_i, C_k), d(C_j, C_k)\Big\}$$

implies

$$\min\Big\{d(C_i, C_k), d(C_j, C_k)\Big\} \leq d(C_i \cup C_j), C_k) \quad \forall\, i, j, k;$$

in words, merging two clusters C_i and C_j less dissimilar between themselves than with another cluster C_k cannot make the resulting dissimilarity with C_k smaller than the smallest initial one. Dissimilarities $D = (d_{k\ell})$ induce a *nearest neighbor* relation, with one or more pairs of reciprocal near neighbors. When the reducibility property holds, each pair of reciprocal near neighbors will be merged before merging with other clusters. Updating chains of nearest neighbors yields a $\theta(N^2)$ agglomerative hierarchical clustering algorithm for the (average) variance criterion [7]. This result extends to the single-linkage, complete-linkage and average-linkage algorithms [98]. When entities of O belong to a low-dimensional Euclidean space and dissimilarities are equal to distances between them, techniques from computational geometry can be invoked, to get even faster algorithms. Extensions of agglomerative hierarchical clustering algorithms to weak hierarchies or pyramids have been much studied recently, e.g., in [2, 9].

3.2 Divisive hierarchical clustering algorithms

Divisive hierarchical clustering algorithms are less frequently used than agglomerative ones. They proceed from an initial cluster containing all entities by successive bipartitions of one cluster at a time until all entities belong to different clusters. Thus, they fit into the following scheme:

Initialization
$P_1 = \{C_1\} = \Big\{\{O_1, O_2, \ldots, O_N\}\Big\}$;
$k = 1$;
Current step:
While $k < N$ **do**
select $C_j \in P_k$ following a first local criterion;
partition C_j into C_{2k} and C_{2k+1} following a second local criterion;
$P_{k+1} = \Big(P_k \cup \{C_{2k}\} \cup \{C_{2k+1}\}\Big) \setminus \{C_j\}$;
$k = k + 1$
EndWhile

The role of the first local criterion is not crucial, as it only determines the order in which clusters will be bipartitioned. The real difficulty lies in bipartitioning the chosen cluster according to the second criterion, a problem which requires a specific algorithm for each case, and which may be NP-hard. Only a few divisive clustering algorithms have, as yet, been proposed.

For the minimum diameter criterion one exploits a property of any maximum spanning tree MST' of the graph G defined above:

Proposition 2 *[53, 97] The unique bicoloring of MST' defines a minimum diameter bipartition of O.*

Note that the diameter of this bipartition is equal to the largest dissimilarity of an edge outside MST' closing an odd cycle with the other edges in MST'.

Using Proposition 2 at all levels yields an $O(N^3)$ divisive hierarchical algorithm [102, 75]. A more careful implementation, building simultaneously maximum spanning trees at all levels, takes $O(N^2 \log N)$ time [55].

It follows from Proposition 2 and the remark following it that there are at most $O(N)$ candidate values for the diameter of a bipartition. This property can be used in a divisive algorithm for hierarchical clustering with the average diameter criterion. Candidate values for the largest diameter are considered in sequence and minimum values for the smallest diameter sought for by dichotomous search. Existence of a bipartition with given diameters is tested by solving a quadratic boolean equation [59] or by a specialized labelling algorithm [97, 46]. The resulting algorithm takes $O(N^3 \log N)$ time. It is more difficult to build an algorithm for average linkage

divisive hierarchical clustering: bipartitioning O to maximize the average between clusters dissimilarity is strongly NP-hard [60]. However, moderate size problems ($N \leq 40$) can be tackled, using hyperbolic and quadratic 0–1 programming. For several criteria, when entities are points in $\mathbb{R}^2$, there are hyperplanes separating the clusters. This property is exploited in an algorithm for hierarchical divisive minimum sum-of-squares clustering in low-dimensional spaces [72] which solves instances with $N \leq 20000$ in $\mathbb{R}^2$, $N \leq 500$ in $\mathbb{R}^3$ and $N \leq 150$ in $\mathbb{R}^4$.

3.3 Global criteria

As mentioned in the previous section, a complete hierarchy of partitions can be viewed as a set of $2N - 1$ clusters. Optimizing an objective function defined on this set of clusters is still unexplored, except for the average split criterion (where the split of O itself is assumed to be 0): the single-linkage algorithm maximizes this value [61].

Results of hierarchical clustering can be represented graphically on a *dendrogram* [23] or an *espalier* [69] as shown in Figure 1. Then vertical lines correspond

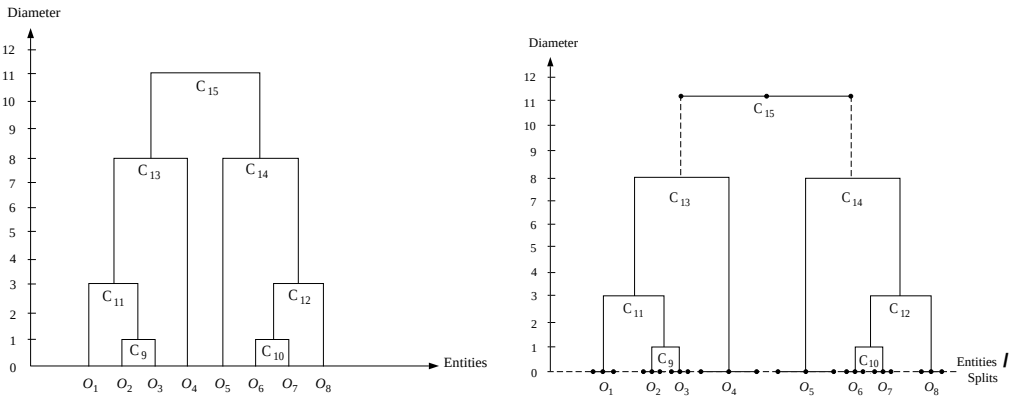

Figure 1: A dendrogram and an espalier, from [69]

to entities or clusters and horizontal lines joining endpoints of vertical lines to mergings of clusters. The height of the horizontal lines corresponds to the value of the updated dissimilarity between the clusters merged. This is a measure of separation or homogeneity of the clusters obtained. In espaliers the length of the horizontal lines is used to represent a second measure of homogeneity or separation of the clusters. If the reducibility condition holds the updated dissimilarities $d'_{k\ell}$ satisfy the ultrametric inequality [88]:

$$d'_{k\ell} \leq \max(d'_{kj}, d'_{j\ell}) \quad \forall\, j, k, \ell.$$

Thus a hierarchical clustering algorithm transforms a dissimilarity $D = (d_{k\ell})$ into an ultrametric $D' = (d'_{k\ell})$. This suggests further criteria: one can minimize

$$\sum_{k,\ell} (d_{k\ell} - d'_{k\ell})^2$$

or

$$\sum_{k,\ell} |d_{k\ell} - d'_{k\ell}|.$$

In the former case, which is NP-hard [86], a combination of the average linkage algorithm with branch-and-bound solves small instances ($N \leq 20$) [17]; in the latter case a branch-and-bound method solves slightly larger instances. Heuristics use penalty methods [30], in which violations of the ultrametric inequality are penalized, or iterative projection strategies [77]. They can be extended to the case where some data are missing [31] and to more general problems discussed in Section 5.

4 Partitioning

4.1 Dynamic Programming

In one-dimensional clustering problems, entities $O_1, O_2, \ldots, O_N$ correspond to points $x_1, x_2, \ldots, x_N$ on the Euclidean line. Such problems are best solved by dynamic programming, e.g., [6, 109]. This method works well when clusters have the *string property*, i.e., consist of consecutive points on the line. Assume $O_1, O_2, \ldots, O_N$ are indexed in order of non-decreasing values of $x_1, x_2, \ldots, x_N$. Let $f(C_j)$ denote the contribution of cluster C_j to the objective function (assumed to be additive in the clusters and to be minimized) and F_m^ℓ the optimal value of a clustering of $O_1, O_2, \ldots, O_m$ into ℓ clusters. The recurrence equation may be written:

$$F_m^\ell = \min_{\{k \in \ell, \ell+1, \ldots, m\}} \left\{ F_{k-1}^{\ell-1} + f(C_m) \right\}$$

where

$$C_m = \{O_k, O_{k+1}, \ldots, O_m\}.$$

Using updating to compute the $f(C_j)$ for all potential clusters yields $O(N^2)$ algorithms for various criteria [102, 109]. Note that the string property does not always hold. Optimal clusters for one-dimensional clique partitioning do not necessarily satisfy it [11]. However, they enjoy a weaker *nestedness property*: let $[C_j]$ denote the range of the entities $O_k, \ldots, O_\ell$ of C_j, i.e., $[x_k, x_\ell]$. Then for any two clusters C_i and C_j in the set of optimal partitions

$$[C_i] \cap [C_j] = \emptyset \text{ or } [C_i] \subseteq [C_j] \text{ or } [C_j] \subseteq [C_i].$$

So, ranges of any two clusters are either disjoint or included one into the other. Exploiting this property leads to a polynomial algorithm for one-dimensional clique partitioning, also based on dynamic programming [70]. A detailed discussion of nestedness and related properties is given in [78].

When clustering entities in higher-dimensional spaces, there does not seem to be an equivalent of the string property. In a few particular cases, the recurrence equation can be extended [81, 38]. Several authors, e.g. [110], have proposed to impose an order on the entities, for instance the order of points on a Peano curve or the order of traversal in a traveling salesman tour, and then to apply dynamic programming to the resulting one-dimensional problem. Such a procedure quickly gives an optimal solution to an approximation of the given problem. Its proximity to the optimal solution of the problem itself depends on the first step, which is somewhat arbitrary.

To obtain an optimal solution in the general case, nonserial dynamic programming [8] must be used. Let F_S^ℓ denote the optimal value of a clustering of the entities of subset S into ℓ clusters. The recurrence relation then becomes

$$F_S^\ell = \min_{\substack{C_\ell \subset S \\ |C_\ell| \leq |S| - \ell + 1}} \left\{ F_{S \setminus C_m}^{\ell-1} + f(C_m) \right\}.$$

Applying this equation takes time exponential in N, so only small sets of entities ($N \leq 20$) may be considered. Sometimes, constraints accelerate the computations, e.g., if all clusters must be small.

4.2 Graph-theoretical algorithms

As mentioned in the previous section, the single-linkage algorithm provides optimal partitions for the split criterion at all levels of the hierarchy. So it is also a $\theta(N^2)$ algorithm for maximizing the split of a partition of O into M clusters. The problem

of maximizing the average split, or the sum-of-splits, of such a partition is related but different. Its solution relies on the following result:

Proposition 3 *[61] Let $\mathcal{C} = \{C_1, C_2, \ldots, C_{2N-1}\}$ denote the set of clusters obtained when applying the single-linkage algorithm to O. Then for all M there exists a partition P_M^* which maximizes the average split and consists solely of clusters of $\mathcal{C}$.*

Consider then the dual graph of the single-linkage dendrogram, as defined in [61]. It is easy to show that any partition of O into M clusters of $\mathcal{C}$ corresponds to a source-sink path with M arcs in that graph. Then, weight the arcs of the dual graph by the splits of the clusters associated with the edges of the dendrogram they cross. Using dynamic programming to find a cardinality constrained longest path yields a partition P_M^* with maximum average split in $\theta(N^2)$ time.

The relationship between graph coloring and finding a bipartition with minimum diameter was also mentioned in the previous section. In fact, this relationship extends to the general case.

Proposition 4 *[22, 58] Let t be the smallest dissimilarity value such that the partial graph $G_t = (V, E_t)$ of G with $E_t = \left\{\{v_k, v_\ell\}; d_{k\ell} \geq t\right\}$ is M-colorable. Then the color classes in any optimal coloring of G_t define a minimum diameter partition of O into M clusters.*

This relationship can be exploited in the reverse direction to show minimum diameter partitioning is NP-hard for $M \geq 3$ [13, 58], and adapted to prove further NP-hardness results [115]. Updating may be used to exploit Proposition 4 efficiently. Consider graph G_t to which is added an edge. If the vertices of this edge do not have the same color, or if local recoloring (e.g., by bichromatic interchange) gives a coloring with no more colors than previously one can proceed to the next graph. When there is some structure in the set O under study, it will be reflected in the graphs G_t, which are easier to color than random ones, and instances with $N \leq 600$ could indeed be solved.

Minimum diameter partitions are not unique. Enumerating them is discussed in [54]. Alternately, one can adapt the coloring algorithm to find a partition minimizing the second largest cluster diameter, subject to the first being minimum, then the third largest and so on [27].

Partitions obtained with the single-linkage algorithm may suffer from the *chaining effect*: dissimilar entities at the ends of a long chain of pairwise similar entities are assigned to the same cluster. Partitions obtained by the coloring algorithm for minimum diameter may suffer from the *dissection effect* [23]: similar entities may be assigned to different clusters. To avoid both effects one may seek compromise solutions, i.e., efficient partitions for the split and diameter criteria. The resulting *bicriterion cluster analysis* algorithm [28] is based on Propositions 1 and 4. To impose a minimum value on the split it suffices to merge the vertices of G at endpoints of successive edges of MST. Then the resulting reduced graph G_R of G can be colored as described above. Splits and diameters of the efficient partitions may be represented graphically on a *diameter-split map*. It can be used to evaluate whether the set O possesses some structure or not and which partitions appear to be the most natural ones. A single efficient partition for a value of M is a good indication.

Some clustering algorithms apply to graphs, which may be viewed as partial graphs G_t as defined above, for a given t. Clusters may then be defined as maximal components with minimum degree at least δ [91]; a $O(N + |E|)$ algorithm provides a hierarchy of packings corresponding to successive values of δ. When clustering points in $\mathbb{R}^2$, geometric properties may be exploited to obtain low-order polynomial algorithms. For instance, minimum average diameter bipartitioning in the plane can be done in $O(n \log^2 n / \log\log n)$ time [74] and minimizing any monotone function of the diameters of an M cluster partition can be performed in $O(n^{5M})$ time [16].

4.3 Branch-and-bound

Branch-and-bound algorithms have been applied, with some success, to several partitioning problems of cluster analysis. Their efficiency depends on sharpness of the bounds used, availability of a good heuristic solution and efficient branching, i.e., rules which improve bounds for all subproblems obtained in a fairly balanced way.

An algorithm for minimum sum-of-squares partitioning [85, 36] exploits bounds based on assignments of entities to clusters already made, and additivity of bounds for separate subsets of entities. It solves problems with $N \leq 120$ and a few well-separated clusters of points of $\mathbb{R}^2$, but its performance deteriorates in higher dimensional spaces. Another algorithm [84], for minimum sum-of-cliques partitioning, uses bounds based on ranking dissimilarities, which are not very sharp. Problems with $N \leq 50$, $M \leq 5$ can be solved.

Better results are obtained when bounds result form solution of a mathematical program. For minimum sum-of-stars partitioning (the M-median problem) the well-known DUALOC algorithm [42] combined with Lagrangian relaxation of the cardinality constraint [57] is very efficient. Problems with $N \leq 900$ are solved exactly and the dimension of the space considered does not appear to be an obstacle.

A variant of the minimum sum-of-cliques partitioning problem arises when one seeks a consensus partition, i.e., one which is at minimum total distance of a given set of partitions [104], distance between two partitions being measured by the number of pairs of entities in the same cluster in one partition and in different clusters in the other. Dissimilarities may then be positive or negative and the number of clusters is not fixed a priori. This problem can be expressed as follows [90]:

$$\text{Minimize} \quad \sum_{k=1}^{N-1} \sum_{\ell=k+1}^{N} d_{k\ell}\, y_{k\ell}$$

subject to:

$$\begin{aligned} y_{k\ell} + y_{\ell q} - y_{kq} &\leq 1 \quad k = 1, 2, \ldots, N-2 \\ -y_{k\ell} + y_{\ell q} + y_{kq} &\leq 1 \quad \ell = k+1, k+2, \ldots, N-1 \\ y_{k\ell} - y_{\ell q} + y_{kq} &\leq 1 \quad q = \ell+1, \ell+2, \ldots, N \end{aligned}$$

and

$$y_{k\ell} \in \{0, 1\} \quad k = 1, 2, \ldots, N-1,\ \ell = k+1, k+2, \ldots, N.$$

where $y_{k\ell} = 1$ if O_k and O_ℓ belong to the same cluster and $y_{k\ell} = 0$ otherwise. Problems with $N \leq 72$ could be solved [90] by applying the revised simplex method to the dual of the continuous relaxation of the above formulation. No duality gap was observed (nor a branching rule specified for the case where there would be one). A direct branch-and-bound approach is proposed in [39]. A first bound equal to the sum of negative dissimilarities is improved upon be using logical relations between the $y_{k\ell}$ variables (or, in other words, exploiting consequences of the triangle inequalities). For instance is variable $y_{k\ell}$ is equal to 1 then for all indices q either both $y_{k\ell}$ and $y_{\ell q}$ are equal to 1 or both are equal to 0 in any feasible solution. If these variables are free, the bound may be increased by

$$\min \Big\{ \max\{d_{k\ell}, 0\} + \max\{d_{\ell q}, 0\}, \max\{-d_{kq}, 0\} + \max\{-d_{\ell q} 0\} \Big\}.$$

Many further consequences are taken into account and the resulting bounds are quite sharp. Instances with $N \leq 158$ could be solved, more quickly than with a cutting-

plane approach, but less quickly than with a combination of heuristic, cutting planes and branch-and-bound (see next subsection).

4.4 Cutting planes

Until recently, few papers of cluster analysis advocated the cutting-plane approach. The minimum sum-of-cliques partitioning problem has attracted the most attention. Therefore, the convex hull H of integer solutions to the problem defined in the previous section is studied.

Proposition 5 [52] *(i) The dimension of H is $N(N-1)/2$;*
(ii) for all k, ℓ $y_{k\ell} \geq 0$ and $y_{k\ell} \leq 1$ are valid inequalities; the former are always facets and the latter never;
(iii) for all k, ℓ, q the triangle inequalities define facets;
*(iv) for every two disjoint subsets U, V of O, the 2-*partition inequality *induced by U, V, i.e.,*

$$y(U:V) - y(U) - y(V) \leq \min\Big\{|U|, |V|\Big\},$$

where $y(U:V)$ denotes the sum of the variables corresponding to pairs of entities one in U and the other in V, $y(U) = y(U:U)$ and $y(V) = y(V:V)$, is valid and a facet if and only if $|U| \neq |V|$.

Several further families of facets are given. These results are used in a cutting plane algorithm [51] to solve instances with $N \leq 158$. It appears that the triangle inequalities suffice in almost all cases. Facets of the polytope obtained when a cardinality constraint is added have also been studied [19].

Recently, cutting planes have been combined with heuristics, relocalization of the best known solution at the origin (which eases the separation problem) and branch-and-bound. [101]. Minimum sum-of-cliques problems of the literature with $N \leq 158$ are solved very quickly.

Cutting-planes were also used in [82] to solve, in moderate time, the auxiliary problem in a column generation approach (see next subsection) to a constrained minimum-sum-of-cuts partitioning problem (called min-cut clustering).

The cutting plane approach does not seem to be easy to adapt to clustering problems with objectives which are not sums of dissimilarities or to problems in which the

number of clusters is fixed. Further work on cutting-planes for clustering or related problem is [19, 20, 43].

4.5 Column generation methods

The generic partitioning problem of cluster analysis may be expressed as a standard partitioning problem, plus one constraint on the number of clusters, by considering all possible clusters, i.e., subsets of O. This gives a number of columns exponential in N:

$$\begin{aligned}
&\text{Min} && \sum_{t=1}^{2^N-1} f(C_t)\, y_t \\
&\text{subject to:} \\
& && \sum_{t=1}^{2^N-1} a_{jt}\, y_t = 1 \qquad j = 1, 2, \ldots, N \\
& && \sum_{t=1}^{2^N-1} y_t = M, \\
&\text{and} \\
& && y_t \in \{0, 1\} \qquad t = 1, 2, \ldots, 2^N - 1,
\end{aligned}$$

and where a_{jt} is equal to 1 if entity O_j belongs to cluster C_t and 0 otherwise. Despite its enormous size this formulation turns out to be one of the most useful. In order to solve this problem one needs (i) to solve efficiently its continuous relaxation and (ii) to proceed efficiently to a branch-and-bound phase in case the solution of the relaxation is not in integers. We discuss these two aspects in turn.

The standard way to solve linear programs with an exponential number of columns is to use column generation [47, 21]. In this extension of the revised simplex method, the entering column is obtained by solving an auxiliary problem, where the unknowns are the coefficients a_j of the column:

$$\begin{aligned}
&\text{Min} && f(C_j) - \sum_{j=1}^{N} a_j\, u_j - u_{N+1} \\
&\text{subject to:} \\
& && a_j \in \{0, 1\} \qquad j = 1, 2, \ldots, N
\end{aligned}$$

where $(u_1, \ldots, u_N, u_{N+1})$ are the dual variables at the current iteration. Difficulty varies depending on the form of $f(C_j)$ as a function of the a_j. For *minimum sum-of-stars clustering* (or the M-median problem), the first clustering problem solved by column generation [44], solving the auxiliary problem is straightforward: for each potential cluster center k in turn set $a_j = 1$ if $d_{kj} < u_j$ and $a_j = 0$ otherwise. If $\sum_{j/a_j=1} (d_{kj} - u_j) - u_{N+1} < 0$ the column so defined is a candidate to enter the basis.

For the capacitated version of this problem the auxiliary problem reduces to a knapsack problem. For the *sum of cliques* problem the subproblem reduces to quadratic 0–1 programming:

$$\text{Min} \sum_{j=1}^{N-1} \sum_{k=j+1}^{N} d_{jk}\, a_j a_k - \sum_{j=1}^{N} a_j u_j - u_{N+1}$$

in 0–1 variables a_j [93, 82, 68]. For the *minimum sum-of-squares* problem, it reduces to a hyperbolic 0–1 program, in view of Huyghens' theorem, which states that the sum of squared distances to the centroid is equal to the sum of squared distances between entities divided by the cardinality of the cluster:

$$\text{Min} \frac{\sum_{j=1}^{N-1} \sum_{k=j+1}^{N} d_{jk}^2\, a_j a_k}{\sum_{j=1}^{N} a_j} - \sum_{j=r}^{N} a_j u_j - u_{N+1}$$

in 0–1 variables. An iterative solution scheme [37] reduces this problem to a sequence of quadratic programs in 0–1 variables. These last problems, as well as other quadratic 0–1 programs discussed above, can be solved by an algebraïc (or variable elimination) method [24], linearisation [113], cutting planes [3] or branch-and-bound [63], possibly exploiting the persistency properties of roof duality theory [56]. Combining column generation with an interior point method [41] allows solution of minimum sum-of-squares partitioning problem with $N \leq 150$.

Once the entering column is found the algorithm proceeds to a simplex iteration as in the revised simplex method. However, convergence may be slow, particularly if there are few clusters in the partition and hence massive degeneracy of the optimal solution. In fact, even when the optimal solution is found many more iterations may be needed to prove its optimality. Columns in the primal correspond to cutting planes in the dual; a good approximation of the dual polytope around the optimal value for the dual is needed, but little information is available about this optimum. A recent

bundle method in the L_1-norm [40] stabilizes the algorithm while remaining within the column generation framework. It gives good results for continuous sum-of-stars clustering in the plane (the multisource Weber problem), instances with $N = 1060$, $M \leq 50$ being solved[62].

Once the linear relaxation of the master problem is solved, one must check for integrality of the solution. For some problems, as minimum sum-of-cliques clustering, it seems to be fairly often the case. Otherwise, branch-and-bound is needed. Extension of standard dual and primal procedures of mixed-integer programming to column generation [71, 66] is only efficient when there are few integers variables. Setting one fractional variable y_t at 1 modifies substantially the problem as all constraints corresponding to elements of C_t are satisfied; but setting y_t at 0 only excludes one column among an enormous number. So other branching rules are needed, and have indeed been found. A first proposal [100] was made in 1983 for capacitated sum-of-stars partitioning (or the capacitated M-median problem with single-supply constraints): branching is done by assigning an entity to a center, which implies this center is selected in some cluster of the partition, or forbidding it to belong to a cluster with that center. Another fairly close branching rule, first proposed [107] for the partitioning problem (but not for column generation) is to specify that two entities O_j and O_k must belong to the same cluster or not. So branching is done in the auxiliary problem by adding the constraints $a_j = a_k$ in one branch, and $a_j + a_k \leq 1$ in the other. Columns not satisfying these constraints are removed. This rule appears to be more efficient than the previous one [67] and variants of it have been applied with success in several papers on scheduling problems, e.g., [33]. Nevertheless, some recent column generation methods for clustering,e.g., [93, 82] still stopped after solution of the master's problem relaxation or used some heuristic from that point. In a recent survey [4], the name "branch-and-price" has been proposed for combination of column generation and branch-and-bound.

4.6 Heuristics

For many criteria, exact solution of large clustering problems is out of reach. So there is room for heuristics. Moreover, finding a good initial solution may be important in column generation (if it is well exploited, i.e., if columns close to those of this solution are used to complete the basis; otherwise beginning with the heuristic solution may slow down the solution process).

Traditional heuristics use exchange of entities between clusters or redefinition of clusters from their centroids. The HMEANS algorithm, e.g. [109], for minimum-sum-of-squares partitioning draws an initial partition at random, then proceeds to best exchanges of entities from one cluster to another until a local minimum is attained. The KMEANS algorithm for the same problem, also draws an initial partition at random then computes the cluster centroïds, assigns entities each to the closest of them and iterates until a local minimum is attained. Both procedures can be repeated a given number of times. They give good results when there are few clusters but deteriorate when there are many. Experiments show that the best clustering found with KMEANS may be more than 50% worse then the best known one.

Much better results have been obtained with metaheuristics, i.e., simulated annealing, Tabu search, genetic search, etc [103]. The recent Variable Neighborhood Search [72] proceeds by local search to a local minimum, then explores increasingly distant neighborhoods of that partition by drawing a perturbation at random and doing again a local search. It moves to a new partition and iterates if and only if a better one than the incumbent is found. Experiments show this procedure is very efficient for approximate solutions of large clustering problems.

5 Other clustering paradigms

5.1 Sequential clustering

Most clustering algorithms give results regardless of whether the given set of entities possesses some structure or not. Moreover, all entities must usually be assigned to some cluster. This disregards the possibility of noise, i.e., entities (possibly all of them) which can only be classified arbitrarily. It may therefore be preferable to consider packing problems instead of partitioning problems. Moreover, one may wish to study clusters one at a time, beginning by the most obvious one, removing its entities and iterating. The so-defined sequential clustering [72] is close to methods of image processing:

Current step
Find clusters $C_k \subset O$ with $|C_k| = k = 1, 2, \ldots, |O|$ which optimize a criterion;
Evaluate the best value k^* of k and the significance of cluster C_{k^*}. If it is significant (different from noise) set $O = O \setminus \{C_{k^*}\}$ and iterate; otherwise stop.

Thus, at each step, a single-cluster parametric clustering problem is solved, and followed by a test based on the distribution of values of the criterion. Some cases are easy: finding a maximum split cluster can be done in $\theta(N^2)$ time in view of Proposition 1, rediscovered in [18]. Finding a minimum radius cluster or a minimum star cluster take $O(N^2 \log N)$ time by ranking dissimilarities. Finding a minimum diameter cluster is NP-hard, as well as finding a minimum clique cluster. The former problem can be solved by reducing it to a sequence of maximum clique problems, and the latter by expressing it as a quadratic knapsack problem. Other geometric criteria are considered in [1] and [25].

5.2 Additive clustering

In addition to finding clusters one may use them to explain dissimilarities (or similarities) between pairs of entities, as proposed in additive clustering [108, 95]. Given a matrix $S = (s_{k\ell})$ of similarities between pairs of entities of O one seeks M overlapping clusters $C_1, C_2 \ldots, C_M$ and corresponding weights $\lambda_1, \lambda_2, \ldots, \lambda_M$ to minimize the sum-of-squares of errors:

$$\sum_{k=1}^{N-1} \sum_{\ell=k+1}^{N} \left(s_{k\ell} - \sum_{j | O_k, O_\ell \in C_j} \lambda_j \right)^2$$

In a variant of that model, one cluster contains all entities. Many heuristics have been proposed for its solution, using various techniques of mathematical programming. If one cluster is considered at a time, in a qualitative factor analysis technique [95], the problem is easier and can be reduced to quadratic or hyperbolic 0–1 programming with a cardinality constraint [64].

5.3 Representing dissimilarities by trees

Consider again the dendrogram obtained by a hierarchical clustering algorithm (see Figure 1). This dendrogram can be viewed as a tree, with vertices associated with the N entities, as well as with the $N - 1$ clusters obtained (and represented by points in the middle of the horizontal lines). Edges join vertices if and only if they are joined by lines of the dendrogram crossing no other vertex. Associating with each edge the length of the corresponding vertical segment in the dendrogram, one observes that the length between the vertex corresponding to O and any vertex associated with a

single entity is a constant. This property may be relaxed. Then the general problem of representing dissimilarities by additive trees arises: the length corresponding to $d_{k\ell}$ will be that of the path between the vertices v_k and v_ℓ of the additive tree T. So both the topology of T and the length of its edges must be determined. This topic is studied in depth in [5].

In order to be representable by an additive tree, it is necessary and sufficient that the dissimilarity D' satisfy the *four-point condition* [14]:

$$d'_{ij} + d'_{k\ell} \leq \text{Max}\,(d'_{ik} + d'_{j\ell}, d'_{i\ell} + d'_{jk}) \quad \forall\, i, j, k, \ell.$$

Finding a dissimilarity D' satisfying this condition and at minimum distance of a dissimilarity D for the minimum sum-of-squares criterion is NP-hard. Indeed, it subsumes the NP-hard problem of finding an ultrametric at minimum distance of a dissimilarity discussed in Section 3. Only very small instances of this problem can be solved exactly, but many heuristics have been proposed. They include generalizations of the penalty approach discussed above [29, 31], iterative projection strategies on closed convex sets defined by the constraints [77], and alternating methods in which local modifications in the tree's topology alternate with fittings of distances to edges [45].

6 Conclusions

Mathematical programming has been applied with success to cluster analysis in the last 25 years. This has led to (i) define precisely many cluster analysis problems, (ii) determine their computational complexity, (iii) clarify the objective underlying known algorithms, and exhibit some important properties, e.g., for the split criterion, (iv) obtain improved and sometimes best possible algorithms for known easy problems; (v) obtain polynomial and sometimes best possible algorithms for new problems, e.g., average split partitioning; (vi) obtain non polynomial but useful algorithm for NP-hard problems, e.g., clique partitioning and minimum sum-of-squares partitioning; (vii) devise useful heuristics, yielding near-optimal solutions for large instances.; (viii) establish ties between cluster analysis and other subfields of mathematical programming and computational geometry, where similar problems are studied.

While many results have been obtained, much remains to be done to completely integrate cluster analysis within mathematical programming. Axiomatics are needed, particularly for partitioning. New exact algorithms should be devised, mostly for divisive hierarchical clustering, sequential clustering and additive clustering, where few or none exist, but also for partitioning with little studied criteria. Heuristics for large instances deserve further study. Empirical comparison of methods is also too rare, with a few exceptions (e.g. [94]). Finally, gathering existing software, often hard to access, and streamlining it into a package would be of help.

References

[1] A. Aggarwal, H. Imai, N. Katoh and S. Suri, Finding k Points with Minimum Diameter and Related Problems, *Journal of Algorithms* **12** (1991) 38–56.

[2] H.J. Bandelt and A.W.M. Dress, Weak Hierarchies Associated with Similarity Measures: an Additive Clustering Technique, *Bulletin of Mathematical Biology* **51** (1989) 133–166.

[3] F. Barahona, M. Junger and G. Reinelt, Experiments in Quadratic 0–1 Programming, *Mathematical Programming* **44** (1989) 127–137.

[4] C. Barnhart, E.L. Johnson, G.L. Nemhauser and M.W.P. Savelsbergh, *Branch and Price: Column Generation for Solving Huge Integer Programs*, Computational Optimization Center COC-94-03, Georgia Institute of Technology, Atlanta, 1994, (revised 1995).

[5] J.-P. Barthelemy and A. Guénoche, *Les Arbres et les représentations des proximités* (Masson: Paris 1988) English translation: *Trees and Proximity Relations* (Wiley: Chichester 1991).

[6] R. Bellman, A Note on Cluster Analysis and Dynamic Programming, *Mathematical Biosciences* **18** (1973) 311–312.

[7] J.P. Benzecri, Construction d'une classification ascendante hiérarchique par la recherche en chaîne des voisins réciproques, *Les Cahiers de l'Analyse des Données* **VII(2)** (1982) 209–218.

[8] U. Bertole and F. Brioschi, *Nonserial Dynamic Programming* (Academic Press: New-York 1972).

[9] P. Bertrand, Structural Properties of Pyramidal Clustering. In: I. Cox, P. Hansen and B. Julesz (eds.) *Partitioning Data Sets* (American Mathematical Society: Providence 1995) 35–53.

[10] J.C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms* (Plenum: New York 1981).

[11] E. Boros and P.L. Hammer, On Clustering Problems with Connected Optima in Euclidean Spaces, *Discrete Mathematics* **75** (1989) 81–88.

[12] P. Brucker, On the Complexity of Clustering Problems. In M. Beckmann and H.P. Kunzi (eds.), *Optimization and Operations Research, Lecture Notes in Economics and Mathematical Systems* **157** (Heidelberg: Springer-Verlag, 45–54, 1978).

[13] M. Bruynooghe, Classification ascendante hiérarchique des grands ensembles de données: un algorithme rapide fondé sur la construction des voisinages réductibles, *Les Cahiers de l'Analyse des Données* **3** (1978) 7–33.

[14] 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) 387–395.

[15] G.L. Leclerc, Comte de Buffon, *Histoire Naturelle, Premier discours: de la manière d'étudier et de traiter l'histoire naturelle* (Paris, 1749).

[16] V. Capoyleas, G. Rote and G. Woeginger, Geometric Clusterings, *Journal of Algorithms* **12** (1991) 341–356.

[17] J.L. Chandon, J. Lemaire and J. Pouget, Construction de l'ultramétrique la plus proche d'une dissimilarité au sens des moindres carrés, *RAIRO-Recherche Opérationnelle* **14** (1980) 157–170.

[18] M.S. Chang, C.Y. Tang and R.C.T. Lee, A Unified Approach for Solving Bottleneck k-Bipartition Problems, *Proceedings of the 19th Annual Computer Science Conference* (San Antonio, Texas, March 5–7, ACM, 1991) 39–47.

[19] S. Chopra and M.R. Rao, On the Multiway Cut Polyhedron, *Networks* **21** (1991) 51–89.

[20] S. Chopra and J.H. Owen, Extended Formulations for the A-Cut Problem, *Mathematical Programming* **73** (1996) 17–30.

[21] V. Chvatal, *Linear Programming* (New-York: Freeman, 1983).

[22] N. Christofides, *Graph Theory. An Algorithmic Approach* (London: Academic Press, 1975).

[23] R.M. Cormack, A Review of Classification (with Discussion), *Journal of the Royal Statistical Society* **A 134** (1971) 321–367.

[24] Y. Crama, P. Hansen and B. Jaumard, The Basic algorithm for Pseudo-Boolean Programming Revisited, *Discrete Applied Mathematics* **29** (1990) 171–185.

[25] A. Datta, H.-P. Lenhof, Ch. Schwarz and M. Smid, Static and Dynamic Algorithms for k-point Clustering Problems, *Journal of Algorithms* **19** (1995) 474–503.

[26] W.H.E. Day and H. Edelsbrunner, Efficient Algorithms for Agglomerative Hierarchical Clustering Methods, *Journal of Classification* **1** (1984) 7–24.

[27] M. Delattre and P. Hansen, Classification d'homogénéité maximum, in: *Actes du Colloque Analyse de Données et Informatique*, INRIA 1, 1977, 99–104.

[28] M. Delattre and P. Hansen, Bicriterion Cluster Analysis, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *PAMI 2* (1980) 277–291.

[29] G. De Soete, A Least Squares Algorithm for Fitting Additive Trees to Proximity Data, *Psychometrika* **48** (1983) 621–626.

[30] G. De Soete, A Least Squares Algorithm for Fitting an Ultrametric Tree to a Dissimilarity Matrix, *Pattern Recognition Letters* **2** (1984) 133–137.

[31] G. De Soete, Ultrametric Tree Representation of Incomplete Dissimilarity Data, *Journal of Classification* **1** (1984) 235–242.

[32] G. De Soete, Additive Tree Representations of Incomplete Dissimilarity Data, *Quality and Quantity* **18** (1984) 387–393.

[33] J. Desrosiers, F. Soumis and M. Desrochers, Routing with Time Windows by Column Generation, *Networks* **14** (1984) 545–565.

[34] E. Diday, From Data to Knowledge: Probabilistic Objects for a Symbolic Data Analysis. In: I. Cox, P. Hansen and B. Julesz (eds): *partitioning Data Sets.* (American Mathematical Society: Providence 1995) 35–53.

[35] E. Diday, *Orders and Overlapping Clusters by Pyramids* (Research Report, 730, INRIA, France 1987).

[36] G. Diehr, Evaluation of a Branch and Bound Algorithm for Clustering, *SIAM Journal on Scientific and Statistical Computing* **6** (1985) 268–284.

[37] W. Dinkelbach, On Nonlinear Fractional Programming, *Management Science* **61** (1995) 195–212.

[38] A. Dodge and T. Gafner, Complexity Relaxation of Dynamic Programming for Cluster Analysis. In: E. Diday *et al.* (eds): *New Approaches in Classification and Data Analysis*. Studies in Classification, Data Analysis, and Knowledge Organization, (1994) 220–227.

[39] U. Dorndorf and E. Pesch, Fast Clustering Algorithms, *ORSA Journal on Computing* **6** (1994) 141–153.

[40] O. du Merle, D. Villeneuve, J. Desrosiers and P. Hansen, Stabilization dans le cadre de la génération de colonnes, *Les Cahiers du GERAD* G–97–08 (1997).

[41] O. du Merle, P. Hansen, B. Jaumard and M. Mladenović, An Interior Point Algorithm for Minimum Sum-of-Squares Clustering, (in preparation).

[42] D. Erlenkotter, A Dual-based Procedure for Uncapacitated Facility Location, *Operations Research* **26** (1978) 1590–1602.

[43] C.C Ferreira, A. Martin, C.C. De Souza, R. Weissmantel and L.A. Wolsey, Formulation and Valid Inequalities for the Node Capacitated Graph Partitioning Problem, *Mahtematical Programming* **74** (1996) 247–266.

[44] R. Garfinkel, A.W. Neebe and M.R. Rao, An Algorithm for the M-median Plant Location Problem, *Transportation Science* **8** (1974) 217–236.

[45] O. Gascuel and D. Levy, A Reduction Algorithm for Approximating a (Non-Metric) Dissimilarity by a Tree Distance, *Journal of Classification* **13** (1996) 129–155.

[46] S. Gelinas, P. Hansen and B. Jaumard, A Labelling Algorithm for Minimum Sum of Diameters Partitioning of Graphs. In: I. Cox, P. Hansen and B. Julesz (eds): *Partitioning Data Sets*, (American Mathematical Society, Providence 1995) 89–96.

[47] P.C. Gilmore and R.E. Gomory, A Linear Programming Approach to the Cutting Stock Problem, *Operations Research* **9** (1961) 849–859.

[48] A.D. Gordon, *Classification: Methods for the Exploratory Analysis of Multivariate Data* (New York: Chapman and Hall, 1981).

[49] A.D. Gordon, A Review of Hierarchical Classification, *Journal of the Royal Statistical Association* **150** (1987) 119–137.

[50] J.C. Gower and G.J.S. Ross, Minimum Spanning Trees and Single Linkage Cluster Analysis, *Applied Statistics* **18** (1969) 54–64.

[51] M. Grötschel and Y. Wakabayashi, A Cutting Plane Algorithm for a Clustering Problem, *Mathematical Programming* **45** (1989) 59–96.

[52] M. Grötschel and Y. Wakabayashi, Facets of the Clique Partitioning Polytope, *Mathematical Programming* **47** (1990) 367–387.

[53] A. Guénoche, *Partitions with Minimum Diameter* (paper presented at the International Federation of Classification Societies Conference, Charlottesville, USA, 1989).

[54] A. Guénoche, Enumération des partitions de diamètre minimum, *Discrete Mathematics* **111** (1993) 277–287.

[55] A. Guénoche, P. Hansen and B. Jaumard, Efficient Algorithms for Divisive Hierarchical Clustering with the Diameter Criterion, *Journal of Classification* **8** (1991) 5–30.

[56] P.L. Hammer, P. Hansen and B. Simeone, Roof Duality, Complementation and Persistency in Quadratic 0–1 Optimization, *Mathematical Programming* **28** (1984) 121–155.

[57] P. Hanjoul and D. Peeters, A Comparison of Two Dual-Based Procedures for Solving the p-Median Problem, *European Journal of Operational Research* **20** (1985) 387–396.

[58] P. Hansen and M. Delattre, Complete-Link Cluster Analysis by Graph Coloring, *Journal of the American Statistical Association* **73** (1978) 397–403.

[59] P. Hansen and B. Jaumard, Minimum Sum of Diameters Clustering, *Journal of Classification* **4** (1987) 215–226.

[60] P. Hansen, B. Jaumard and E. da Silva, Average-Linkage Divisive Hierarchical Clustering, *Les Cahiers du GERAD*, G–91–55 (1991). To appear in *Journal of Classification*.

[61] P. Hansen, B. Jaumard and O. Frank, Maximum Sum-of-Splits Clustering, *Journal of Classification* **6** (1989) 177–193.

[62] P. Hansen, B. Jaumard, S. Krau and O. du Merle, A Column Generation Algorithm for the Weber Multisource Problem (in preparation).

[63] P. Hansen, B. Jaumard and V. Mathon, Constrained Nonlinear 0–1 Programming, *ORSA Journal on Computing* **5** (1993) 97–119.

[64] P. Hansen, B. Jaumard and C. Meyer, Exact Sequential Algorithms for Additive Clustering, Les Cahiers du GERAD, (1997) (forthcoming).

[65] P. Hansen, B. Jaumard and N. Mladenovic, How to Choose k Entities Among N. In: I. Cox, P. Hansen and B. Julesz (eds.) *Partitioning Data Sets* (American Mathematical Society, Providence 1995) 105–116.

[66] P. Hansen, B. Jaumard and M. Poggi De Aragão, Mixed-Integer Column Generation Algorithms and the Probabilistic Maximum Satisfiability Problem. In: E. Balas, G. Cornuejols and R. Kannan (eds.) *Proceedings Second IPCO Conference*, (Carnegie-Mellon University, 1992), 165–180.

[67] P. Hansen, B. Jaumard and E. Sanlaville, Weight Constrained Minimum Sum-of-Stars Clustering, *Les Cahiers du GERAD*, G–93–38, (1993). To appear in *Journal of Classification*.

[68] P. Hansen, B. Jaumard and E. Sanlaville, Partitioning Problems of Cluster Analysis: A Review of Mathematical Programming Approaches. In: E. Diday *et al.* (eds) *New Approaches in Classification and Data Analysis* (Springer: Berlin 1994) 228–240.

[69] P. Hansen, B. Jaumard, B. Simeone, Espaliers, A Generalization of Dendrograms, *Journal of Classification* **13** (1996) 107–127.

[70] P. Hansen, B. Jaumard, B. Simeone, Polynomial Algorithms for Nested Univariate Clustering, *Les Cahiers du GERAD*, G–96–28, (1996).

[71] P. Hansen, M. Minoux and M. Labbe, Extension de la programmation linéaire généralisée au cas des programmes mixtes, *Comptes Rendus de l'Académie des Sciences*, Paris, 305 (1987) 569–572.

[72] P. Hansen, N. Mladenovic, Variable Neighborhood Search, *Les Cahiers du GERAD* G–96–49 (1996). To appear in *Computers and Operations Research*.

[73] J.A. Hartigan, *Clustering Algorithms* (New York: Wiley, 1975).

[74] J. Hershberger, Minimizing the Sum of Diameters Efficiently, *Computational Geometry: Theory and Applications* **2** (1992) 111–118.

[75] L.J. Hubert, Some Applications of Graph Theory to Clustering, *Psychometrika* **39** (1974) 283–309.

[76] L.J. Hubert, Min and Max Hierarchical Clustering Using Asymmetric Similarity Measures, *Psychometrika* **38** (1973) 63–72.

[77] L.J. Hubert and P. Arabie, Iterative Projection Strategies for the Least-Squares Fitting of Tree Structure to Proximity data, *British Journal of Mathematical and Statistical Psychology* **48** (1995) 281–317.

[78] F.K. Hwang, U.G. Rothblum and Y.-C. Yao, *Localizing Combinatorial Properties of Partitions*, AT&T Bell Labs Report, (1995).

[79] M. Jambu, *Classification automatique pour l'analyse des données, Tome 1* (Dunod: Paris 1976).

[80] M. Jambu, *Exploratory and Multivariate Data Analysis* (Academic Press: New York 1991).

[81] R.E. Jensen, A Dynamic Programming Algorithm for Cluster Analysis, *Operations Research* **17** (1969) 1034–1057.

[82] E.L. Johnson, A. Mehrotra and G.L. Nemhauser, Min-cut Clustering, *Mathematical Programming* **62** (1993) 133–151.

[83] L. Kaufman and P.J. Rousseeuw, *Finding Groups in Data: An Introduction to Cluster Analysis* (New York: Wiley, 1990).

[84] G. Klein and J.E. Aronson, Optimal Clustering: A Model and Method, *Naval Research Logistics* **38** (1991) 447–461.

[85] W.L.G. Koontz, P.M. Narendra and K. Fukunaga, A Branch and Bound Clustering Algorithm, *IEEE Transactions on Computers* **C–24** (1975) 908–915.

[86] M. Krivanek and J. Moravek, NP-Hard Problems in Hierarchical-Tree Clustering, *Acta Informatica* **23** (1986) 311–323.

[87] G.N. Lance and W.T. Williams, A General Theory of Classificatory Sorting Strategies. 1. Hierarchical Systems, *The Computer Journal* **9** (1967) 373–380.

[88] B. Leclerc, Description combinatoire des ultramétriques, *Mathématiques et Sciences Humaines* **73** (1981) 5–37.

[89] J.K. Lenstra, Clustering a Data Array and the Traveling Salesman Problem, *Operations Research* **22** (1974) 993–1009.

[90] J.F. Marcotorchino and P. Michaud, *Optimisation en analyse ordinale des données* (Masson: Paris 1979).

[91] D.W. Matula and L.L. Beck, Smallest-Last Ordering and Clustering and Graph-Coloring Algorithms, *Journal of the Association for Computing Machinery* **30** (1983) 417–427.

[92] W.T. McCormick Jr, P.J. Schweitzer and T.W. White, Problem Decomposition and Data Reorganization by a Clustering Technique, *Operations Research* **20** (1972) 993–1009.

[93] M. Minoux and E. Pinson, Lower Bounds to the Graph Partitioning Problem through Generalized Linear Programming and Network Flows, *RAIRO–Recherche Opérationnelle* **21** (1987) 349–364.

[94] G.W. Milligan and M.C. Cooper, An Examination of Procedures for Determining the Number of Clusters in Data Set, *Psychometrika* **50** (1985) 159–179.

[95] B. Mirkin, Additive Clustering and Qualitative Factor Analysis Methods for Similarity Matrices, *Journal of Classification* **4** (1987) 7–31, (Erratum 6, 271–272).

[96] B. Mirkin, *Mathematical Classification and Clustering* (Kluwer: Dordrecht 1996).

[97] C. Monma and S. Suri, Partitioning Points and Graphs to Minimize the Maximum or the Sum of Diameters. In: Y. Alavi, G. Chartrand, O.R. Oellerman, A.J. Schwenk, eds., *Graph Theory, Combinatorics, and Applications, Proceedings of the Sixth Quadrennial International Conference on the Theory and Applications of Graphs* (New York: Wiley, 1991) 899–912.

[98] F. Murtagh, A Survey of Recent Advances in Hierarchical Clustering Algorithms, *The Computer Journal* **26** (1983) 329–340.

[99] J. Ponthier, A.-B. Dufour and N. Normand, *Le modèle Euclidien en analyse des données* (Ellipses: Paris 1990).

[100] A.W. Neebe and M.R. Rao, An Algorithm for the Fixed-Charge Assignment of Users to Sources Problem, *Journal of the Operational Research Society* **34** (1983) 1107–1113.

[101] G. Palubeckis, A Branch-and-Bound Approach using Polyhedral Results for a Clustering Problem, *INFORMS Journal on Computing* **9** (1997) 30-42.

[102] M.R. Rao, Cluster Analysis and Mathematical Programming, *Journal of the American Statistical Association* **66** (1971) 622–626.

[103] C.R. Reeves, (ed.) *Modern Heuristic Techniques for Combinatorial Problems* (Blackwell: London, 1993).

[104] S. Régnier, Sur quelques aspects mathématiques des problèmes de classification, *ICC Bulletin* **4** (1965) 175–191, reprinted in *Mathématiques et Sciences Humaines* **82** (1983) 85–111.

[105] P. Rosenstiehl, L'arbre minimum d'un graphe. In: P. Rosenstiehl (ed.): *Théorie des Graphes* (Paris, Dunod, 1967) 357–368.

[106] A. Rusch and R. Wille, Knowledge Spaces and Formal Concept Analysis. In: H. Boch and W. Polarek (eds) *Data Analysis and Information Systems* (Springer: Berlin 1996) 427–436.

[107] D.M. Ryan and B.A. Foster, An Integer Programming Approach to Scheduling. In: A. Wren (ed.), *Computer Scheduling of Public Transport Urban Passenger Vehicle and Crew Scheduling* (North-Holland: Amsterdam 1981) 269–280.

[108] R.N. Shepard and P. Arabie, Additive Clustering Representation of Similarities as Combinations of Discrete Overlapping Properties, *Psychological Review* **86** (1979) 87–123.

[109] H. Späth, *Cluster Analysis Algorithms for Data Reduction and Classification of Objects* (Ellis Horwood, Chichester, 1980).

[110] L.E. Stanfel, A Recursive Lagrangian Method for Clustering Problems, *European Journal of Operational Research* **27** (1986) 332–342.

[111] P.H.A. Sneath and R.R. Sokal, *Numerical Taxonomy* (Reeeman: San Francisco 1973).

[112] R.E. Tarjan, An Improved Algorithm for Hierarchical Clustering Using Strong Components, *Information Processing Letters* **17** (1983) 37–41.

[113] F. Vanderbeck, *Decomposition and Column Generation for Integer Programs*, Ph.D. Thesis, Faculté des Sciences Appliquées, Université Cahtolique de Louvain, Louvain-la-Neuve, 1994.

[114] H.D. Vinod, Integer Programming and the Theory of Grouping, *Journal of the American Statistical Association* **64** (1969) 506–519.

[115] W.J. Welch, Algorithmic Complexity — Three NP-hard Problems in Computational Statistics, *Journal of Statistical Computing* **15** (1982) 68-86.