

1986

The Topological Information Retrieval System and the Topological Paradigm: a Unification of the Major Models of Information Retrieval.

Steven C. Cater
Louisiana State University and Agricultural & Mechanical College

Follow this and additional works at: https://digitalcommons.lsu.edu/gradschool_disstheses

Recommended Citation

Cater, Steven C., "The Topological Information Retrieval System and the Topological Paradigm: a Unification of the Major Models of Information Retrieval." (1986). *LSU Historical Dissertations and Theses*. 4289.

https://digitalcommons.lsu.edu/gradschool_disstheses/4289

This Dissertation is brought to you for free and open access by the Graduate School at LSU Digital Commons. It has been accepted for inclusion in LSU Historical Dissertations and Theses by an authorized administrator of LSU Digital Commons. For more information, please contact gradetd@lsu.edu.

INFORMATION TO USERS

While the most advanced technology has been used to photograph and reproduce this manuscript, the quality of the reproduction is heavily dependent upon the quality of the material submitted. For example:

- Manuscript pages may have indistinct print. In such cases, the best available copy has been filmed.
- Manuscripts may not always be complete. In such cases, a note will indicate that it is not possible to obtain missing pages.
- Copyrighted material may have been removed from the manuscript. In such cases, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, and charts) are photographed by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each oversize page is also filmed as one exposure and is available, for an additional charge, as a standard 35mm slide or as a 17"x 23" black and white photographic print.

Most photographs reproduce acceptably on positive microfilm or microfiche but lack the clarity on xerographic copies made from the microfilm. For an additional charge, 35mm slides of 6"x 9" black and white photographic prints are available for any photographs or illustrations that cannot be reproduced satisfactorily by xerography.

8710552

Cater, Steven C.

THE TOPOLOGICAL INFORMATION RETRIEVAL SYSTEM AND THE
TOPOLOGICAL PARADIGM: A UNIFICATION OF THE MAJOR MODELS OF
INFORMATION RETRIEVAL

The Louisiana State University and Agricultural and Mechanical Col. Ph.D. 1986

University
Microfilms
International 300 N. Zeeb Road, Ann Arbor, MI 48106

**THE TOPOLOGICAL INFORMATION RETRIEVAL SYSTEM
AND THE TOPOLOGICAL PARADIGM:
A UNIFICATION OF THE MAJOR MODELS
OF INFORMATION RETRIEVAL**

A Dissertation

Submitted to the Graduate Faculty of the
Louisiana State University and
Agricultural and Mechanical College
in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

in

The Department of Computer Science

by
Steven C. Cater
B.S., Louisiana State University, 1978
M.S., Louisiana State University, 1981
December 1986

Acknowledgement

I wish to thank my major professors, Duncan A. Buell, Donald H. Kraft, and John A. Hildebrant, without whom this work could not have been done; and my wife, Kathryn Gallagher, without whom this work would not have been done.

Table of Contents

| | |
|---|-----|
| Acknowledgement | ii |
| Table of Contents..... | iii |
| List of Figures | iv |
| Abstract | v |
| I. Introduction..... | 1 |
| II. The p -Norm Model, and Some Questions | 6 |
| III. The TIRS Model | 17 |
| IV. Model Unifications | 53 |
| V. Conclusion | 62 |
| Bibliography | 66 |
| Appendix - A Review of Algebra and Topology..... | 71 |
| Vita..... | 84 |
| Approval Sheet | 85 |

List of Figures

| | | |
|-----|--------------------------------------|----|
| 1. | A Sample Document Space | 25 |
| 2. | A Type One Query, Part One..... | 27 |
| 3. | A Type One Query, Part Two..... | 28 |
| 4. | A Type One Query, Part Three..... | 29 |
| 5. | A Type Two Query, Part One..... | 32 |
| 6. | A Type Two Query, Part Two | 33 |
| 7. | A Type Two Query, Part Three | 34 |
| 8. | A Type Three Query, Part One..... | 38 |
| 9. | A Type Three Query, Part Two | 39 |
| 10. | A Type Three Query, Part Three | 40 |
| 11. | A Type Four Query, Part One..... | 41 |
| 12. | A Type Four Query, Part Two..... | 42 |
| 13. | A Type Four Query, Part Three..... | 43 |

Abstract

There are three topics discussed in this work. The first topic is an investigation of the topological properties of the p -norm model of Salton, Fox, and Wu. It is shown that certain properties of the p -norm model that one would expect to hold, given the topological origin of the model, do not in fact hold. These properties include the ability to change the query by changing p , and the ability to adequately separate documents. Since these properties do hold in the model as actually constructed, it must be that the properties do not follow from the topological origin of the model.

The second topic is a search for a usable model with an adequate theoretical basis. In order to construct such a model, the topological paradigm is defined. This paradigm establishes a minimal set of requirements that any system with a topological foundation should have. A particular example of the paradigm, the Topological Information Retrieval System (TIRS), is constructed. It is shown that all of the desired properties of the p -norm model hold for the TIRS model. A discussion of the various query systems that may be used with TIRS is given. These query systems include a natural language interface and a weighted boolean query system, as well as two specialized interfaces. The weighted boolean query system has the property that <attribute, weight> pairs, when treated as units, have all of the properties of the non-weighted boolean lattice. The run time of the system is estimated, once for an inverted file implementation, and once for an implementation using kd-trees. These run times are much better than for traditional systems.

The third topic is a reexamination of the standard models of information retrieval, considered as cases of the topological paradigm. The paradigm is shown to be a unifying model, in that all of the standard models, i.e., the boolean, vector space, fuzzy set theoretic, and probabilistic models, as well as a hierarchical model, are shown to be instances of the paradigm.

An appendix contains a review of relevant topics from topology and abstract algebra.

Chapter One

Introduction

The problem of information retrieval can be stated quite simply: given a collection of data, determine which of the data possess a given set of properties. There are several methods currently utilized to make these determinations. If the data can be placed into a fixed, relatively unchanging format, then it is possible to construct a data base system around the data. If it possible to determine which properties form the major distinctions in the data, then it may be possible to construct a fixed or slowly changing classification scheme for the data. Many expert systems have this basic structure.

A different problem arises if the data are of varied format, are constantly changing in number and value, and if the major properties of the collection are impossible to determine in advance. It is this last problem that is the subject area of information retrieval, as it will be considered in this work. There are at least two instances of this problem which must be addressed. The first, and more important, is the question and answer system. In this system, there is a large collection of data, which is constantly being updated, and one is allowed to ask questions of the system, which the system must answer, using the available data. There are no working question and answer systems, as defined above. Indeed, it may be that the first working question and answer system will pass the Turing test. An easier problem is the document retrieval problem. This problem is important, both as an adjunct to bibliographic research, and as a small domain for a question and answer system. Hence, this work will interpret information retrieval as document retrieval.

It is possible to formally define the information retrieval problem. Several definitions exist in the literature [Kra85, Rad83a]. The definition to be used here follows [1].

Definition: An information retrieval system is an ordered octuple

$$IR = \langle DOC, QY, DS, QS, RS, f_d, f_q, f_r \rangle,$$

where

DOC is a finite set of *documents* to be used as data in the information retrieval system,

QY is the set of all possible natural language *queries* that may be asked of the system,

DS is the *document space*, the abstract document representation used in the matching of documents to queries,

QS is the *query space*, the abstract query representation used in the matching process,

RS is the *relevance space*, the set of values that indicates the degree of correspondence between documents and queries,

$f_d: DOC \rightarrow DS$ is the *indexing* or *abstraction function*, which maps the documents into the document space,

$f_q: QY \rightarrow QS$ is the *translation function*, which maps the natural language queries into the query space, and

[1] Cf [Rad83a].

$f_r: DS \times QS \rightarrow RS$ is the *formal relevance function*, the mapping that associates the documents with the queries.

Currently, there are five major approaches to information retrieval discussed in the literature. These approaches are:

- 1° the boolean model, as exemplified by commercial systems such as DIALOG and BRS [Sal83c],
- 2° the standard model (sometimes called the “vector space” model), exemplified by SMART [Sal83c],
- 3° the fuzzy set theoretic, or generalized boolean, model [Bue82, Bue81a, Bue81c, Rad83a, Rad83b, Rad84],
- 4° the probabilistic models [Sal83c, Coe82, Mar60, Kra78, Boo74], and
- 5° the hierarchical model [Kol83].

There seems to be a general agreement that not all of these models are actually different; indeed, it has already been shown that the two primary models of probabilistic information retrieval are actually the same [Rob82]. More recently, Salton, Fox, and Wu [Sal83b] have added p -norms to the basic vector space model, raising the possibility that the vector space and the fuzzy set theoretic approaches are the same. The current state of affairs is this:

- 1° The boolean model is a special case of the vector space, the fuzzy set theoretic, and the probabilistic models, but not the hierarchical model.
- 2° There is one basic version of each of the vector space, the fuzzy set theoretic, and the probabilistic models.

- 3° The p -norm version of the vector space model holds the promise of unifying the fuzzy set theoretic and the vector space models.
- 4° The hierarchical model seems to be quite different from each of the other models; there has been little or no work on a unification of this model with the others.

In this work, an attempt is made to unify all five of these competing models. The unifying model, called the topological paradigm, can be understood most naturally as a generalization of the p -norm vector space model, and will be introduced and discussed in that context. Chapter Two of this work introduces the p -norm model, discusses its many advantages, and illustrates what seem to be two problems with the model. In Chapter Three a particular implementation of the topological paradigm is constructed. This model, called the Topological Information Retrieval System (TIRS), is shown to retain all of the advantages of the p -norm model, while not sharing its apparent problems. The four main types of queries and users of a TIRS system are examined, and examples given of each. Also in this chapter, asymptotic estimates of the run time of the TIRS model are given, which, if confirmed by experiment, will allow the TIRS model to operate faster than other systems of information retrieval.

Chapter Four is concerned with the unification proper. The demonstrations given in this chapter show that each of the other models can be considered as an instance of the topological paradigm, and hence not too different from the TIRS model. Finally, Chapter Five provides an overview of the results of the earlier chapters, and includes a discussion of possible future work. Included here are suggestions for experiments which could settle some of the questions inherent in the TIRS model, and an outline of

some of the steps necessary in the transformation of TIRS from a theoretical model into a commercial system.

Since the topological paradigm is a new approach to information retrieval, there may be certain mathematical terms and concepts which are unfamiliar. As an aid to those who desire a review of these topics, an appendix is included, in which the algebraic and topological terms used in this work are defined and explained.

Chapter Two

The p -Norm Model, and Some Questions

The p -Norm Model

Since the TIRS model is best understood as a generalization of the p -norm model, it is necessary to review the construction of the p -norm model. This discussion will follow the original presentation in Salton, Fox, and Wu [Sal83b]. As is usual in a vector space model, a document is described as a series of possibly orthogonal attributes, which are usually weighted to give a vector space of unit intervals. Let D be a document having term weights d_A and d_B , respectively, for attributes A and B . If one wishes to find the similarity of D to a query of the form $Q_{A \text{ and } B}$, then clearly it is better for document D to be about *both* A and B , rather than about one but not the other. Salton, Fox, and Wu claim that one should measure similarity in a situation such as this by measuring distance from (1,1) (the point which is most about both A and B), “[i]t is clear that for **and**-queries, the (1, 1) point on the map, representing the situation where both items are present in an item, is the desired situation.” [2] Similarly, if one is interested in the query $Q_{A \text{ or } B}$, then it is better to be farther from (0,0) rather than closer; “. . . for **or**-queries, on the other hand, the (0, 0) point identifying the situation where both terms are absent from an item is to be avoided.” [3]

Hence, in a two term query, similarity measures can be given by

$$\text{sim}(D, Q_{A \text{ or } B}) = \sqrt{\frac{d_A^2 + d_B^2}{2}} \quad (1)$$

[2] [Sal83b], page 1023.

[3] [Sal83b], page 1023.

and

$$\text{sim}(D, Q_A \text{ and } B) = 1 - \sqrt{\frac{(1-d_A)^2 + (1-d_B)^2}{2}} \quad (2)$$

where the equations in (1) and (2) are obtained from the Euclidean distance formulas in 2-space.

Once the basic idea of similarity as distance has occurred, there is nothing to prevent the generalizations to more than two attribute terms, to the use of weighted query terms, and to the use of other than standard Euclidean distance measures: "The basic similarity measurements . . . can be extended to the case of *weighted-query terms*" [4], and, "[t]he notion of distance between points in a document space can be generalized by introducing the well-known L_p vector norm defined for an n -dimensional vector (d_1, d_2, \dots, d_n)" [5] If these generalizations are made, then one has the definition of the p -norm model of information retrieval [6].

Definition: Let

$$\{A_1, A_2, \dots, A_n\} \quad (3)$$

be a set of attribute terms. Define a document D as a tuple of term weights,

$$D = (d_{A_1}, d_{A_2}, \dots, d_{A_n}). \quad (4)$$

A "generalized Boolean or-query," i.e., one using p -norms and query term weights, can be written as

$$Q_{or(p)} = \left[(A_1, a_1) \text{ or}^p (A_2, a_2) \text{ or}^p \dots \text{ or}^p (A_n, a_n) \right] \quad (5)$$

[4] [Sal83b], page 1024.

[5] [Sal83b], page 1024.

[6] [Sal83b], pages 1024 - 1025.

where a_i is the query weight associated with attribute term A_i . Similarly, a "generalized Boolean and-query" is given by

$$Q_{and(p)} = \left[(A_1, a_1) \text{ and}^p (A_2, a_2) \text{ and}^p \cdots \text{and}^p .ne \text{ } 4.00i (A_n, a_n) \right] \quad (6)$$

Finally, the similarities between document D and queries $Q_{or(p)}$ and $Q_{and(p)}$ are given by

$$sim(D, Q_{or(p)}) = \left[\frac{a_1^p d_{A_1}^p + a_2^p d_{A_2}^p + \cdots + a_n^p d_{A_n}^p}{a_1^p + a_2^p + \cdots + a_n^p} \right]^{\frac{1}{p}} \quad (7)$$

and

$$sim(D, Q_{and(p)}) = 1 - \left[\frac{a_1^p (1-d_{A_1})^p + a_2^p (1-d_{A_2})^p + \cdots + a_n^p (1-d_{A_n})^p}{a_1^p + a_2^p + \cdots + a_n^p} \right]^{\frac{1}{p}} \quad (8)$$

Note that the definitions above are based on the idea of p -norms, but are not true distances, since the query weights and the document weights are multiplied rather than subtracted. As in the 2-space case, the distance involved is distance from $(0, 0, \dots, 0)$ and from $(1, 1, \dots, 1)$, which is a specialized notion of distance that will prove to be too specific.

Several claims are made for this model. The ones that will be a concern in this work are

- 1* "When $p = \infty$ and the query and document term weights are limited to 0 or 1, a conventional Boolean retrieval model is produced." [7] Thus, the p -norm model includes the boolean model as a special case.

[7] [Sal83b], page 1025.

- 2° “[W]hen $p = 1$, the distinction between the **and** and **or** connectives in a query disappears and a simple vector-processing retrieval model is obtained. . . .” [8]
The p -norm model also includes the standard vector space model.
- 3° “[W]hen $p = \infty$ and the query is unweighted, the query-document matching function is dependent only on the document term of highest weight for Q_{or} and the document term of lowest weight for Q_{and} . This is precisely the situation . . . mentioned for the fuzzy-set model of retrieval. . . .” [9] The authors claim to include the fuzzy set theoretic model as another special case.
- 4° “The interpretation of the query structure can be altered by using different p -values to compute the query-document similarity.” [10] By this the authors seem to mean that as the p -values change, the meaning of the resulting numbers changes.
- 5° An extended p -norm system can be built on top of an inverted file system. “A complete iterative search procedure using the extended Boolean retrieval system, but compatible with the established conventional retrieval environments based on inverted file systems may then be specified” [11] Since most systems in existence are of the inverted file type, this feature is of decided utility.

In a later chapter, it will be shown that the TIRS model retains all of the actual advantages of the p -norm model.

[8] [Sal83b], page 1025.

[9] [Sal83b], page 1025.

[10] [Sal83b], pg. 1028.

[11] [Sal83b], page 1035.

Unfortunately, it does not seem to be the case that all of the above claims hold for the p -norm model; in particular, claim 4° seems problematic. In addition, there appears to be another disadvantage of the p -norm model: in an unweighted or fixed weight system, there exist distinct documents that cannot be adequately separated (as defined later) by the p -norm model. These two problems will be considered below.

Changing p -Values Does Not Change the Interpretation

Note that in the definition of query-document similarity stated above, the explicit assumption of the finiteness of the set of attributes was made. This assumption results in the use of a finite dimensional Euclidean space as the document space, with the *origin*

$$\sum_{i=1}^{i=n} X\{0\} \quad (9)$$

and *identity*

$$\sum_{i=1}^{i=n} X\{1\} \quad (10)$$

as the points from which the similarity distances are computed. There are two claims to be shown here.

Claim 1: Topologically, the space that results from allowing weighted-query terms is homeomorphic [12] to that resulting from not allowing such terms.

Proof: We will show that the space resulting from a fixed set of weighted-query terms is the same space as that resulting from a set of terms which are all 1. Let X be the n -dimensional topological space resulting from measuring objects based on a

[12] Definitions of topological and algebraic terms are given in the appendix.

similarity query with weights

$$A = \{a_1, a_2, \dots, a_n\}; \quad (11)$$

without loss of generality we may assume that none of the a_i are zero, since these coordinates will have no part in either of the similarity calculations. Let Y be the n -dimensional topological space resulting from measuring objects based on a similarity query with unit (or no) weights. Now, the topology of each of these spaces is fully described by giving a subbasis for each topology. But note that, in each case, the subbasis is the set of points which can be made distinct by some combination of similarity measures from the origin and the identity. In other words, two points l and m are in different sets if

$$\text{dist}(l, \prod_{i=1}^{i=n} \{1\}) \neq \text{dist}(m, \prod_{i=1}^{i=n} \{1\}) \quad (12)$$

or if

$$\text{dist}(l, \prod_{i=1}^{i=n} \{0\}) \neq \text{dist}(m, \prod_{i=1}^{i=n} \{0\}) \quad (13)$$

or both; the points are in the same element of the subbasis if and only if both distances are the same.

Consider now a point $x \in X$. We may assume that x has coordinates

$$x = (x_1, x_2, \dots, x_n). \quad (14)$$

Note that a point $y \in Y$ having coordinates

$$y = \left[\frac{x_1}{a_1^p}, \frac{x_2}{a_2^p}, \dots, \frac{x_n}{a_n^p} \right] \quad (15)$$

will have the same distance in Y from the origin and identity as x will have in X ; it is this point y that we will identify with x . Hence our proposed homeomorphism

$f : X \rightarrow Y$ is given by

$$f(x) = \left[\frac{x_1}{a_1^p}, \frac{x_2}{a_2^p}, \dots, \frac{x_n}{a_n^p} \right]. \quad (16)$$

Now f is clearly continuous and invertible, and so is a homeomorphism. Hence the two spaces are homeomorphic. \square [13]

Claim 2: Changing the choice of p in the norm does not change the topology.

Proof: The proof of this claim is similar to the proof of Claim 1. We will show that, over a non-weighted space, any p -norm space is homeomorphic to the corresponding 1-norm space. As a result, then, any two p -norm non-weighted spaces are equivalent; application of Claim 1 then finishes the proof.

Fix p , and consider the space X resulting from using distinct sets of points as measured from the origin and identity in a fixed n -dimensional space (as in Claim 1). As before, what is needed is a homeomorphism mapping X into the space Y described by 1-norms. Let $f : X \rightarrow Y$ be defined by

$$f(x) = (x_1^{1/p}, x_2^{1/p}, \dots, x_n^{1/p}) \quad (17)$$

Then f is a homeomorphism between X and Y . \square

One possible argument that might be made against the results claimed in these two proofs is an argument of the form, "What we are concerned with is not the topology of the space, but rather the form of the query; if we change the measurement of the query, we must be changing things." Unfortunately for those who would make this argument, it seems to be invalid. Note that a query is nothing more than a

[13] Perhaps a simpler way of seeing this claim is as follows: Each space is a product space of intervals. All intervals are homeomorphic. The product of homeomorphisms is a homeomorphism. \square

distance from the origin or the identity. We have shown that the change of norm or weight in a query does not affect the topology; since distance is a topological property, a change of query that results in a topologically equivalent space is not really a change at all. What is changing when one changes the weights or the p -norms in the above topology is not the points themselves, but rather only the particular numbers associated with the points. Since the numbers associated with the points were chosen arbitrarily to begin with, a change of either of the types described above cannot change the topology.

Thus, the first point has been made: changing the p -norms does not change anything real. Note that more has actually been shown, since it turns out that there is really only one topology under discussion. We will assume in the remainder of this chapter, unless stated otherwise, that we are using a system with no weights and with $p = 1$.

The Space Does Not Adequately Separate Points

This point has been hinted at in the earlier discussion; it will now be considered in detail. We make the following definition:

Definition: Two points (documents) in an information retrieval system are *non-separable* if there is no query which returns exactly one of the points.

It will be shown that, for a fixed system of weights, that no query can separate certain points in a p -norm system. The easiest example occurs in the two dimensional case. Let the 2-space unit cube (square) be the document space, and consider Euclidean distance ($p = 2$). Note that this space is symmetric with respect to our simi-

larity measurements about the two coordinates. That is, let

$$l = (a, b) \quad (18)$$

and

$$m = (b, a) \quad (19)$$

be points in the unit cube. Then

$$\text{sim}(l, Q_{A \text{ or } B}) = \text{sim}(m, Q_{A \text{ or } B}) \quad (20)$$

and

$$\text{sim}(l, Q_{A \text{ and } B}) = \text{sim}(m, Q_{A \text{ and } B}), \quad (21)$$

for all queries Q . [14] As a result, points (like the two above) which are on the intersection of an *and* level curve and an *or* level curve are non-separable.

Non-separability does not depend either on the use of 2-space or on Euclidean distance. Indeed, since the definition of similarity in the general case is

$$\text{sim}(D, Q) = \left[\frac{\sum_{i=1}^{i=n} a_i^p \hat{d}_i^p}{\sum_{i=1}^{i=n} a_i^p} \right]^{\frac{1}{p}} \quad (22)$$

where

$$\hat{d}_i = \begin{cases} d_i & \text{if } Q = \text{or} \\ (1-d_i) & \text{if } Q = \text{and} \end{cases} \quad (23)$$

and finite summation is commutative, it must be that there are non-separable points for any finite dimension n . This is indeed the case. In general, the set of non-separable points is homeomorphic to S^{n-2} , i.e., the $n-2$ dimensional unit circle [15],

[14] The symmetry is not as easy to see if weights are allowed, but it is still there (for each fixed set of weights).

[15] Note that S^n lives in $n+1$ dimensional space, and that isometric "lines" in n dimensions are of dimension $n-1$.

so there may actually be an infinite number of non-separable points. This is an unfortunate occurrence, and one that intuitively should not be allowed to happen. There should be ways of separating different documents.

A proponent of the p -norm model might interject at this point, "But wait . . . if you give me any two points, then I can supply a set of query weights and a p -norm such that the two points are separated." This is a true statement, but note that a similar true statement can be made about a system with a fixed value of p . Since the weights associated with the attribute terms are themselves somewhat arbitrary, it is difficult to see how adding another set of arbitrary numbers to a measurement can add any information. Further, this query illustrates that the desired origin of the system as a vector space does not carry through to the particular model, since points are separable in the original vector space, but, after the similarity measure is defined, points may only be separated by using query weights. At best, it seems that a dilemma exists within the p -norm model.

The Nature of the p -Norm Problems

The two situations discussed above arise from the same basic problem. The notion of similarity as defined by Salton, Fox, and Wu begins with the idea of distance in a metric space. However, by the time the definition is complete, the notion of similarity has been changed from an actual distance (which must be defined for all pairs of points, not for all single points and one special pair of points) to something that is not nearly as strong (since most of the pairs in the original distance definition are no longer used). As an example of just how weak the resulting space is, one can consider

the standard separation axioms of topology [Dug66]. Under this system, spaces are placed into six classes: T_0 through T_4 , and non- T_0 . In general, we have $T_{i+1} \subseteq T_i$, so that the T_0 spaces are the most inclusive. A metric space, the type of space that allows the proper definition of distance, belongs to the most restrictive type, T_4 . This was the type of space that would have resulted if the notion of distance had been used throughout. But in the previous section it was shown that there exist points which cannot be separated in a subbasis. This result shows that the space as defined by Salton, Fox, and Wu cannot be T_4 ; actually, the space is not even T_0 . Most topologists do not work with spaces which are not at least T_2 , perhaps feeling that such spaces are too ‘wild’, i.e., have too many non-intuitive properties.

There is a clear need for a better model, one based on the idea of distance between points, as in this model, but one which looks more like Euclidean space. The topological paradigm and the TIRS model, both described in the next chapter, are the result of the search for such a model.

Chapter Three

The TIRS Model

In this chapter, the TIRS model will be formally constructed. Using the basic ideas of the p -norm, a topologically adequate space for the representation of documents and queries will be found, followed by the embedding of the document set into the space. It will then be demonstrated that this model avoids the problems inherent in the p -norm model, while retaining its advantages. We begin by considering some of the properties that are desired in the TIRS model.

The Topological Paradigm - A Basis for TIRS

There are several properties that any distance based model should possess. We consider the major ones below.

- 1° As recognized by Salton, Fox, and Wu, the notion of distance should be central to an information retrieval model. This property requires that the underlying document space be one in which it is possible to find the distance between any pair of points, and not just distances from special points. As a result, then, one is forced to use a metric space as the underlying document space.
- 2° A document must be represented by a set of attribute terms; hence the space must be a product space. This is already a standard practice, but the property is mentioned here since it forms the basis of the representation method. Also, each document must be capable of being described using only a finite number of attribute terms. This property is also standard. We do not require, however, that the total number of attribute terms be finite, as required by most other systems. Note

that the requirement of a fixed number of attribute terms in a system is a disadvantage if one is going to allow automatic indexing, since these systems use term frequencies based on the text of the document. Salton, Fox and Wu are ambiguous on this point in their description of the p -norm model: the definitions are given in terms of a fixed number of attribute terms; but the experiments described, with indexing based on term frequencies, seem to allow for varying numbers of attribute terms. We will make the specific assumption that there is available an infinite but countable collection of attribute terms.

- 3^{*} Each attribute term must be weighted with an appropriate totally ordered set. Most of the available systems tend to use sub-intervals of the real line. In the TIRS model, we will use the standard interval $[0, 1]$, but there is nothing to prevent the use of other totally ordered sets, such as $[-1, 1]$, or the boolean $\{0, 1\}$. Note that as a result of 2^{*}, only a finite number of terms may be non-zero. This requirement makes sense, as most would agree that a particular document may only be "about" a finite number of things. [16]
- 4^{*} A query is a description of a hypothetical "perfect" document. If the query were described exactly as is a document, then one would have a standard "vector space" model. Most systems, however, recognize that there is a tradition of handling a query differently from a document. Queries are normally described by means of boolean operators, with term weights (and occasionally operator weights). There is a good reason for this when interpreted in the light of our

[16] Most (all?) systems already make this assumption.

model: it might be the case that more than one document is "perfect". We therefore extend our query definition: a query is a description of a finite set of "perfect" documents. This extension will allow the use of standard query formulations, and has the following property: A query is nothing more than a finite set of points in the document space, with each point in the set representing a possible "perfect" document.

This set of properties forms the axioms of the *topological paradigm*; it is this paradigm that will be shown to be the unifying feature of all of the standard models as well as the TIRS model. Two things should be noted here. The first is these axioms are actually weaker than those used in a standard "vector space" model, since several features of a vector space are missing. Some of the missing features are the addition and multiplication operators which are a part of the definition of vector space, the assumption of orthogonality of the coordinates (not a part of the definition, but present in most systems), and the requirement that the metric be a standard one (generally one inherited from the reals). The second object of note is that these axioms only define the theoretical portions of the system, and hence ignore the very real problems of initial query formulation, document entry, and the other problems inherent in converting a real world problem into a form amenable to theory. In particular, these axioms do not require that any particular translation system be used; only that an adequate one be used. In the next section, the TIRS model is constructed. For the construction, we will assume that an adequate translation system is available, and later discuss particular types of systems that might be used.

The TIRS Construction

As a result of the axioms above, one is obligated to require that the document space be an infinite dimensional metric space. The space DS that will be used is a countably infinite space with

$$DS = \prod_{i=1}^{\infty} [0, 1] \quad (24)$$

A document D will be a point in DS , again subject to the axioms: D can have only a finite number of non-zero terms. We will denote an arbitrary element from DS as

$$D = (d_1, d_2, \dots, d_n, 0, \dots), \quad (25)$$

where the

$$\{d_i\}_{i=1}^n \quad (26)$$

are each elements of $[0, 1]$, with d_n being the last non-zero element in the point.

Note that DS contains points which are not allowable as documents; the point $(1, 1, 1, \dots)$ is such a point. This point, if it could represent something, would be about everything, which is not allowed. [17] One could argue that the system would be better defined by allowing only points which can represent documents, i.e., by defining DS as the set

$$\bigoplus_{i=1}^{\infty} [0, 1] = \left\{ (d_1, d_2, \dots, d_k, 0, \dots) \mid d_i \in [0, 1], k < \infty \right\}. \quad (27)$$

This system could be used, since it will have all of the needed properties. The advantages to be gained from using the original definition are mostly in ease of description; e.g., our space is a product space of weights, whereas the alternate space is a subspace

[17] However, the point $(0, 0, 0, \dots)$ is a valid document point.

under the subspace topology of a product space of weights. It is true that our space has properties (e.g., completeness) not possessed by the restricted space; however, no use is made of any of these properties. As a result, one could change to the more restrictive definition with no loss of necessary properties.

As mentioned in the axioms, a query must be treated as a finite set of document descriptions. Each query is processed in the same general fashion, as follows:

- 1° Construct the projection space having only the coordinates (attributes) given in the query. Thus, even though the space is an infinite dimensional one, each query is answered in a finite dimensional space.
- 2° Locate the query in the projection space.
- 3° Using the metric, find the documents closest to the query.
- 4° Return the documents, ranked according to smallest distance from the query.

Note that the exact form of the query is not fixed by the above requirements. We will look at three forms the query may take in the projection space, and define the *retrieved set* for each. This will be followed by specific examples of what seem to be the four major types of queries.

Query Forms, and Retrieved Sets

The first, and simplest, case is that of a query consisting of a single point in DS . This is already a fairly complex situation, however, as that single point represents a perfect document of an arbitrarily large (but finite) number of attribute terms, each with its own term weight. There are two ways of interpreting this query: as a vector space model query, and (equivalently) as a weighted boolean query containing only

AND operators. Either of these will locate a single point in DS . In order to retrieve documents, the idea of *clause weights* is introduced. The use of a clause weight is similar to the choice of p in the p -norm model; it is a weighting associated with a clause [18] which affects the manner in which documents are retrieved. In the p -norm model, this clause weight is the p -norm itself; in the TIRS model, it is an actual distance [19], representing the cutoff beyond which a document is said to be far away. Hence, the retrieval process for this case is as follows: given the query

$$Q = (q_1, q_2, \dots, q_k, 0, \dots) \quad (28)$$

and the clause weight w , locate the point Q in DS , then retrieve all documents which are a distance of w or less away from Q . The documents are presented in order of increasing distance from Q , giving control over the number of documents presented, and, with luck, presenting the most useful documents first. The *retrieved set* of documents for query Q , R_Q , is defined in this case as

$$R_Q = \{x \mid \text{dist}(Q, x) \leq w\}. \quad (29)$$

The second case to be considered is that of a query set containing more than one element, say j elements (points), i.e.,

$$Q = \{Q_1, Q_2, \dots, Q_j\} \quad (30)$$

[18] Or clause operator, in the p -norm model.

[19] The TIRS model that will be discussed in this section is one based on term and clause weights, i.e., one based on a metric space with the space defined in terms of *balls* of varying radius. It is possible to move the clause weights in with the term weights in the TIRS model, which would then have a weight and a distance cutoff for each coordinate. This latter model is based on a metric space with the space considered as the product of each coordinate space, under the product topology. These two interpretations of the TIRS model give the same topology, and so are not really different. This "product" interpretation will be used in the discussion of TIRS sophisticated users and queries.

where each

$$Q_i = (q_1, q_2, \dots, q_k, 0, \dots), \quad (31)$$

as before. We associate with each element of the query a clause weight, which may be obtained either by default, by distributing a single weight to each query point, or by having a separate weight for each point. Let w_i be the weight of query point Q_i ; the retrieved set for this type of query is then

$$R_Q = \{x \mid (\exists i \in 1..j) \text{ dist}(Q_i, x) \leq w_i\} \quad (32)$$

A document D is to be retrieved if it is sufficiently close to one of the query points; the retrieved set will be presented in the order of absolute distance from the appropriate point, so that the first document presented is the one that was closest overall, the second was second closest overall, not necessarily to the same point, etc. The boolean equivalent of this type of set is a boolean query using *AND* and *OR* which is expressed in disjunctive normal form, with weights associated with each disjunct. Since any non-contradiction can be expressed in disjunctive normal form, this query form is expressive enough to work for any boolean query.

The third type of query is a slight generalization of the last type. It may be that one of the points is more important to the query than is another. This can be accounted for by having a second type of weight associated with a clause (point), an *ordering* weight. This ordering weight will not affect the retrieval of items, so the retrieved set is the same as in the previous case, but the weight will affect the ranking of items on the output. The ordering weight is multiplied by the distance from the query point, and the result is used to order the documents. This allows for relative ranking of points in the query.

With the three types of weights mentioned above, one might be wondering exactly how they interact. It is not necessary for all of these weights to be used; in fact, the use of any one is independent of the use of any other. However, since the use of term weights is a basic aspect of the TIRS model, this work will only consider those query methods having term weights other than from the set $\{0, 1\}$. In the next section four types of queries will be described, representing four distinct types of users that the system could expect.

Users and Examples

In this section we will assume that there are four basic groups of users: naive, traditional experienced, TIRS experienced, and TIRS sophisticated. The existence of a different query type for each group allows the same system to serve the needs of many. Each group will be defined, and an explanation of the query type for that group will be given. In addition, two examples will be given for each query type, the first illustrating a sample query and its translation, and the second a series of figures showing exactly how the documents are retrieved for a query of that type. Figure One is a sample document space of two dimensions. Each of the later figures will use this document space as its starting point.

The naive user is one with no experience with any information retrieval system. Using a system not too different from the standard vector space methods [Sal83a], it is possible to take a query expressed in English and transform it into a single point in the document space. The term weights can be obtained either from the query itself, or by allowing term weights to be placed on all keywords in the query. Such a query system

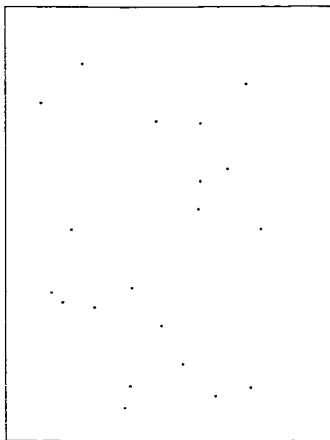


Figure 1. A Sample Document Space.

will result in queries which are always a single point in the document space, and documents can be retrieved using either a standard (default) relevance ball, or the size of the relevance ball may be requested.

As an example, consider the following query.

What information do you have on canasta, bridge, or gin rummy?

Using the heuristic that words mentioned first are more important than words mentioned later, and removing semantically null words, gives the set of <attribute, weight> pairs

$$\{ \langle \text{canasta}, 0.8 \rangle, \langle \text{bridge}, 0.5 \rangle, \langle \text{gin rummy}, 0.3 \rangle \} \quad (33)$$

As mentioned above, this query is similar to one that is used in standard vector space retrieval systems, and can be considered to be a logical conjunction of the terms. In the TIRS system, this query is transformed into a single point

$$(0.5, 0.8, 0.3, 0, \dots) \quad (34)$$

assuming that the attribute terms are stored in lexicographic order as strings, and that the first three attribute terms are the ones in the query. This point is placed into the document space. If there was no relevance ball specified for the query, then the default ball will be used. All documents that are within its radius from the query point will be retrieved, and presented in order of increasing distance from the query. As another possibility, the system can simply begin to retrieve documents in order of increasing distance from the query point, with the user being allowed to stop the process when enough documents have been retrieved.

Figures Two through Four illustrate the process of finding relevant documents in the sample space. Figure Two is the sample document space, with the addition of a single query point, indicated by a plus sign. This is the situation after the query translation has occurred, and the point has been placed in the document space. In Figure Three, the relevance ball of standard radius has been placed around the query point. Finally, in Figure Four the points within the relevance ball have been retrieved, in the indicated order.

The traditionally experienced user is one who is familiar with standard boolean queries, and is able to associate with each attribute term a term weight. The input query for this type of user will be exactly what is expected: a collection of weighted

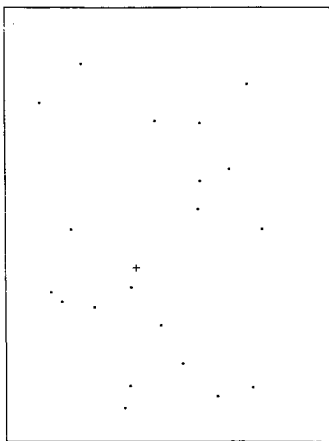


Figure 2. A Type One Query, Part One.

attribute terms, connected by the binary boolean operators *AND* and *OR*, as well as the unary boolean operator *NOT*. [20] The following algorithm will be used to change the input query into a set of points in the document space.

- 1° (Consistency check) Check whether the query is a contradiction. If it is, then no answer is possible, so return an error condition.
- 2° (Convert to DNF) If the query is not a contradiction, then it has a disjunctive normal form. Convert the query into its disjunctive normal form.

[20] The binary operator *NOT* can be expressed using the *AND NOT* combination.

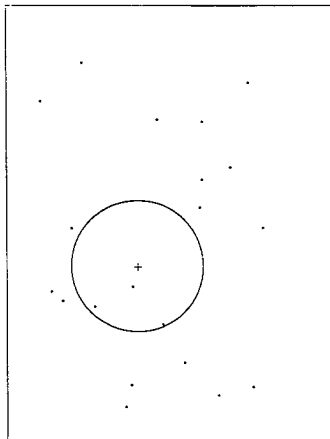


Figure 3. A Type One Query, Part Two.

- 3° (Replace *NOT*'s) Each atom in the disjunctive normal form consists of a single weighted attribute term, either alone or prefixed by a *NOT* operator. Replace each atom of the form *NOT* ($A_i, weight$) by the atom ($A_i, 1 - weight$).
- 4° (Locate the points) The modified DNF of the original equation is now in a form that can be placed in the document space, with each clause forming a point. Place the points in the space.
- 5° (Assign ball radii and retrieve) Associate with each point obtained from the query a relevance ball. As in the earlier example, the radii for the balls may be

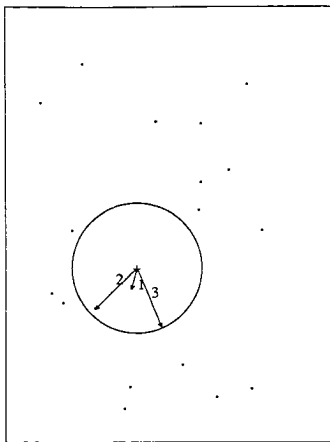


Figure 4. A Type One Query, Part Three.

obtained in several ways. Once the balls have been assigned, simply retrieve the points within the balls.

As an example of this algorithm, consider the following query.

$$\begin{aligned} & NOT (<shoes, 0.1> OR <ships, 0.2>) \\ & \quad AND \\ & (NOT <sealing wax, 0.3> AND \\ & (<cabbages, 0.4> OR NOT <kings, 0.5>)) \end{aligned}$$

Following the above algorithm, a check is first made to determine consistency. Since the query is not a contradiction, it is converted to DNF, resulting in the following expression [21].

[21] The algorithm being used here does not result in a minimal DNF, i.e., one with a minimal number of disjuncts. Rather, it has one disjunct for each combination of atoms which will make

(NOT <shoes, 0.1> AND NOT <ships, 0.2> AND
 NOT <sealing wax, 0.3> AND NOT <cabbages, 0.4> AND
 NOT <kings, 0.5>)

OR

(NOT <shoes, 0.1> AND NOT <ships, 0.2> AND
 NOT <sealing wax, 0.3> AND <cabbages, 0.4> AND
 NOT <kings, 0.5>)

OR

(NOT <shoes, 0.1> AND NOT <ships, 0.2> AND
 NOT <sealing wax, 0.3> AND <cabbages, 0.4> AND <kings, 0.5>)

The next step is to replace the negated atoms by their new values, which results in

(<shoes, 0.9> AND <ships, 0.8> AND
 <sealing wax, 0.7> AND <cabbages, 0.6> AND
 <kings, 0.5>)

OR

(<shoes, 0.9> AND <ships, 0.8> AND
 <sealing wax, 0.7> AND <cabbages, 0.4> AND
 <kings, 0.5>)

OR

(<shoes, 0.9> AND <ships, 0.8> AND
 <sealing wax, 0.7> AND <cabbages, 0.4> AND <kings, 0.5>)

There are three disjuncts, so there will be three points to be located in the document space. Assuming once again that attribute terms are stored in lexicographic order (cabbage, kings, sealing wax, ships, shoes), and that the terms in the query are the first few in the attribute list, we obtain the points

$$\left\{ \begin{array}{l} \langle 0.6, 0.5, 0.7, 0.8, 0.9, 0, \dots \rangle, \\ \langle 0.4, 0.5, 0.7, 0.8, 0.9, 0, \dots \rangle, \\ \langle 0.4, 0.5, 0.7, 0.8, 0.9, 0, \dots \rangle \end{array} \right\} \quad (35)$$

At this point, one assigns relevance balls to each of the points and retrieves all the documents inside these balls.

the original expression true. The advantages of this form are two: the form is unique (unlike a minimal DNF), and no loss of information can occur in the transformation (through the elimination of atoms via the law of the excluded middle). Cf. Cater, Iyengar, and Fuller [Cat84].

Figures Five through Seven illustrate the interpretation of weighted boolean queries. As before, in Figure Five the original document space is given, with the addition of the two points obtained from the query. Note that, since this is only a two dimensional space, only two query points are used, as opposed to the three points obtained in the earlier example. This is not necessary restriction, but it gives a more natural example, since most queries would not have the same attribute listed with multiple (distinct) weights. Figure Six continues the example by placing a relevance ball of standard radius about each query point; Figure Seven shows the retrieved set of documents, listed as before in the actual order of retrieval.

Since there has been some question previously about the ability of vector space systems to use full boolean queries [22], the following theorem will show that they are available in the TIRS system.

Theorem: Let Δ be the set of all finite combinations of valid document points in DS , i.e.,

$$\Delta = \left\{ \left\{ (d_1, d_2, \dots, d_k, 0, \dots)_l \right\}_{l=1}^n \mid d_i \in [0, 1], k < \infty, n < \infty \right\} \quad (36)$$

Let Q be the set of boolean queries having only a finite number of terms, with each term having a term weight in the range $[0, 1]$. Then Δ is the set-homomorphic image of Q [23].

Proof: We will define the homomorphism f by means of the following algorithm [24]:

[22] See, e.g. page 1023 of Salton, Fox, and Wu [Sal83b]

[23] See, e.g., Lang [Lan70], or the appendix, for definitions of the algebraic terms.

[24] Cf. Cater, Iyengar, and Fuller [Cat84], page 139.

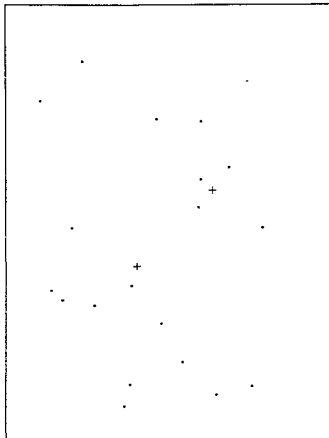


Figure 5. A Type Two Query, Part One.

- 1° Temporarily remove the term weights by assigning a name to each distinct combination of (attribute, weight) pairs. What remains is a boolean query.
- 2° Express the boolean query in DNF by constructing the truth table of the query. Then, for each line in the table which has an overall truth value of true, construct a conjunction consisting of each variable in the original query, prefixed by *NOT* iff the value of that variable in that line is false. The DNF is a disjunction of these conjunctions.

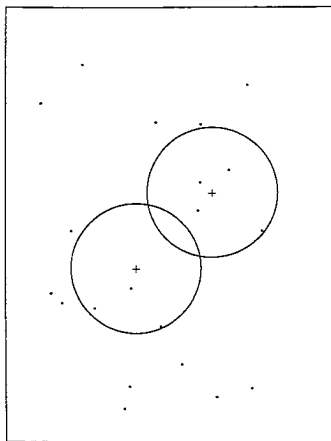


Figure 6. A Type Two Query, Part Two.

- 3° If the DNF exists, then replace the variables by the original (attribute, weight) pairs. Using the algorithm given earlier, associate the DNF of the query with an element of Δ . If the DNF does not exist, assign the empty set as the value of the function.

Note that this algorithm does give a function $f: Q \rightarrow \Delta$, since it assigns a unique value to each element of Q . It remains to show that f is a set-homomorphism, i.e., that f is one to one modulo the appropriate relation, and that f is onto.

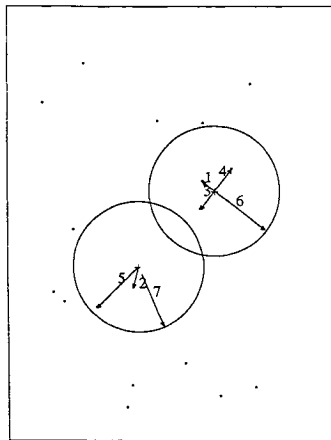


Figure 7. A Type Two Query, Part Three.

To show that f is onto, let $d \in \Delta$. Then d consists of a finite number of points in DS . For each point, construct a conjunction of the non-zero values in the point, using as terms the pairs (attribute, value). Finally, join the conjunctions with *OR* operators, resulting in a single boolean expression. Since there can be only a finite number of non-zero values in each point, and since there were only a finite number of points, the resulting boolean expression must have only a finite number of terms. Hence the expression is in Q .

For the other half of the proof, note that the relation that one should mod out by

is logical equivalence. If queries a and b are logically equivalent, then they will have the same DNF if the above algorithm is used. As a result,

$$f(a) = f(b). \quad (37)$$

Similarly, if a and b are not logically equivalent, then there must be at least one set of truth functional value assignments which makes a and b have different values. This corresponds to distinct lines in the truth tables of a and b , which means that

$$f(a) \neq f(b). \quad (38)$$

As a result of these two observations, it must be that f is a set-homomorphism. \square

Since the homomorphism exists between the two sets, it follows that

- 1° any finite set of points in DS can be accessed through a weighted boolean query,
- 2° the use of boolean queries is consistent, i.e., two queries refer to the same set of points if and only if they are logically equivalent.

Hence, the traditionally experienced user will find in the TIRS model a system which works in a familiar manner. Note that this proof also answers the need expressed by the "wish list" of Waller and Kraft [Wal79, Bue81c, Kra83], in that it states exactly which requirements of a boolean algebra may be maintained by an information retrieval system; in this case, all of the algebra is retained, if one assumes that two <attribute, weight> pairs which differ in either coordinate represent distinct atoms. Bartschi [Bar85] has approached the query evaluation problem in a fashion similar to that done in this proof; his results, which are defined in terms of similarity homomorphisms between queries and documents, are similar to the purely algebraic parts of the TIRS system, but without the ability to utilize other query forms.

Once the user becomes familiar with the TIRS model, he will find that it offers more power than the traditional system. If one restricts the queries allowed to those already in DNF, then one gains the ability to manipulate the document space in additional ways. The TIRS experienced user will have the ability to specify the relevance ball radii and the ordering weights, if desired. This should allow for greater recall from the system, and, since the radii are adjustable, one can seek to maximize the precision-recall product by such adjustments. As an example of the queries utilized by the TIRS experienced user, consider the following.

$$\begin{aligned}
 &(\langle \text{bridge}, 0.5 \rangle \text{ AND } \langle \text{canasta}, 0.3 \rangle, 1.7, 0.2) \\
 &\quad \text{OR} \\
 &\quad (\langle \text{chess}, 0.4 \rangle, 0.4, 1.0) \\
 &\quad \text{OR} \\
 &(\langle \text{ghosts}, 0.7 \rangle \text{ AND } \langle \text{superghosts}, 0.8 \rangle \text{ AND } \langle \text{botticelli}, 0.2 \rangle, 1.0, 0.8)
 \end{aligned}$$

Each disjunct is an ordered triple, consisting of the disjunct of weighted attribute terms, the clause weight, and the ordering weight. Since this query is already in disjunctive normal form, it can easily be converted to a set of points in DS . Assuming once again that the attribute terms given are the first few in DS , and that they are listed in alphabetical order, the translation to points becomes

$$\left\{ \begin{array}{l} (\langle 0, 0.5, 0.3, 0, \dots \rangle, 1.7, 0.2), \\ (\langle 0, 0, 0, 0.4, 0, \dots \rangle, 0.4, 1.0), \\ (\langle 0.2, 0, 0, 0, 0.7, 0.8, 0, \dots \rangle, 1.0, 0.8) \end{array} \right\} \quad (39)$$

There are three points in the query set, corresponding to the three clauses in the original query. The first point, corresponding to the card game clause, has a clause weight of 1.7 and an ordering weight of 0.2. A point is to be retrieved if it is less than a distance of 1.7 from this first point. Each point that is retrieved will have the distance from its query point multiplied by 0.2. This process will be repeated for the other two

points, using clause and ordering weights of 0.4 and 1.0, and 1.0 and 0.8, respectively. Each of the points that was retrieved will then be ranked by its (corrected) distance from its matched query point, and the the documents will be presented in the resulting order. Note that, for ordering weights, a *smaller* value gives a higher rank, since the ordering weight is a distance multiplier.

Figures Eight through Ten illustrate the document retrieval process for the TIRS experienced case. Note that, once again, only two points are given in the transformed query. Also, since the ordering weight is actually a local rescaling of the metric, which is not representable graphically, an ordering weight of one is assumed. The Figures are similar to the earlier traditionally experienced case, except that in Figure 9 the relevance balls have different radii.

It might seem that the TIRS experienced query form is more restricted than that of the traditionally experienced query form, since here one must enter the query in DNF, but in the earlier situation it was possible to use any weighted boolean query. As an answer to this question, one should note that, first, no query is lost by restricting to DNF, since DNF exists for any query which is not a contradiction. As a corollary of the theorem in the last section, the set Δ is isomorphic to the set of boolean queries expressed in DNF (via an application of the first isomorphism theorem for sets); hence, even though a restricted form is all that is allowed here, it is sufficient to ask any possible question. In addition, since the TIRS experienced user has control over the ordering weights and the relevance ball radii, this form of the TIRS model allows additional control over the number and form of the elements in the retrieved sets. This allows for a form of clause weight.

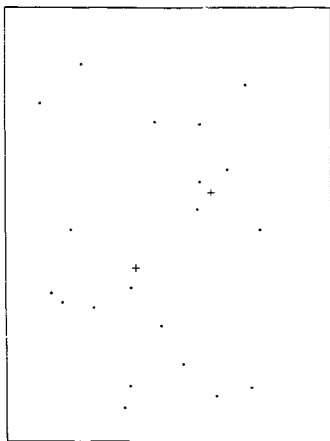


Figure 8. A Type Three Query, Part One.

The last category, that of the TIRS sophisticated user, is similar to the previous category, but allows the query to contain a *relevance width* associated not with each point in *DS*, but with each dimension of each point. Using the example from above, modified to allow for relevance widths, would result in the following query.

```

(<bridge, 0.5, [0.4, 0.65]> AND <canasta, 0.3, [0.25, 0.33]>, 0.2)
OR
(<chess, 0.4, [0.35, 0.45]>, 1.0)
OR
(<ghosts, 0.7, [0.65, 0.75]> AND <superghosts, 0.8, [0.75, 1]>
AND <botticelli, 0.2, [0.1, 0.22]>, 0.8)

```

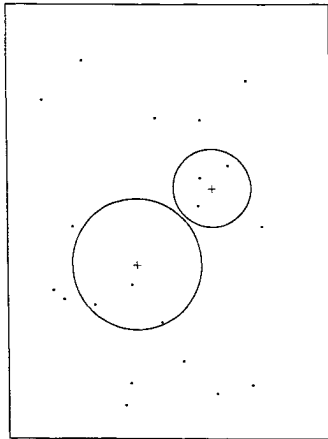


Figure 9. A Type Three Query, Part Two.

In this example, the relevance ball is determined coordinatewise; it is the cross product of the range given in each <attribute, value, range> triple. In the second term, the relevance ball will range from 0.35 to 0.45, as opposed to the earlier range of 0 to 0.8. It is still necessary to have each attribute weight given, as the distances are computed from this point, as before. It is only after the points have been found that the relevance ball comes into use; those documents which are within the relevance ball are retrieved, in order of increasing distance from the point.

Figures Eleven through Thirteen illustrate the query process for a TIRS sophisti-

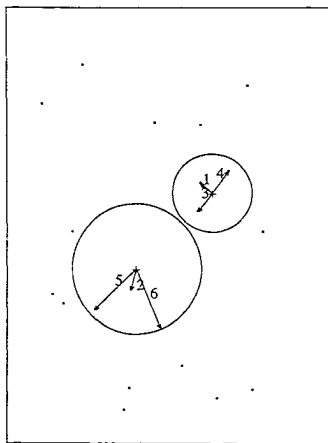


Figure 10. A Type Three Query, Part Three.

cated user. Note the change in the shape of the relevance balls, from the discs of the earlier examples to the rectangles of arbitrary length and width in this case. [25]

Note that, in going from the TIRS experienced user to the TIRS sophisticated user, the only change that has been made is the replacement of the single relevance radius with a relevance interval in each weighted attribute term. This change, which allows the specification of other than balls of fixed radius for the relevance

[25] Actually, the earlier figures have been a bit more specific than they appeared to be, since they were based on the p_2 metric (normal Euclidean distance). The use of any metric is allowed in TIRS; as will be shown later, some are computationally better than others.

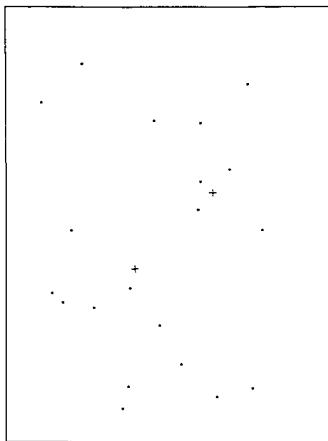


Figure 11. A Type Four Query, Part One.

neighborhoods, may be useful in the following instance: There is a well known problem with having users specify weights in a query, since in general the users will have no knowledge of the weight distribution in the document space. By allowing the specification of ranges, the user will be able to construct queries without this knowledge and still have a reasonable expectation of obtaining good recall. In addition, the use of ranges in each coordinate allows one to partially simulate the threshold fuzzy set theoretic system of Buell and Kraft [Bue81b], by having atoms of the form $\langle \text{attribute}, x, [x, 1.0] \rangle$.

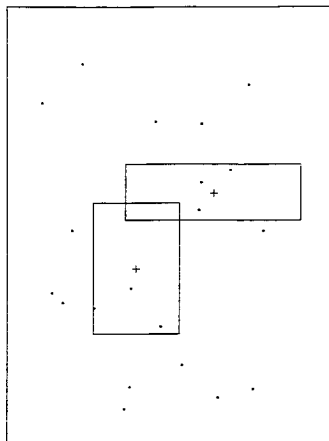


Figure 12. A Type Four Query, Part Two.

With this description of the various interfaces available to a user of the TIRS system, the overview of the model is complete. It remains to show the advantages of the system, and establish a few implementation details.

Advantages of TIRS

Several advantages for the TIRS model have been claimed. Each of these advantages will be considered, and it will be shown that, in each case, the claimed advantage does indeed hold.

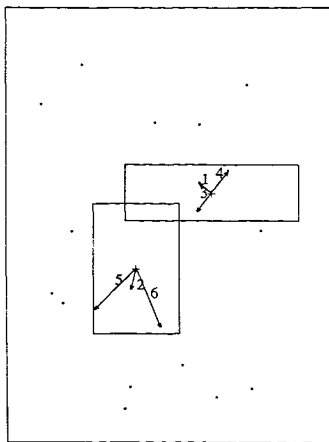


Figure 13. A Type Four Query, Part Three.

The first major source of claims was the set first presented by Salton, Fox, and Wu. [26]

- 1* The size of the output obtained in response to a given query is easy to control. This is controlled in the TIRS model by ranking the output, and by establishing cutoff distances beyond which no searching is done.
- 2* The output is ranked; the assumption of equal importance of retrieved documents does not hold.

[26] Items 1* through 4* are based on the list on page 1022, [Sal83b].

- 3° Term weights, as well as clause and ordering weights, are available. This allows varying levels of importance to be placed on differing parts of the query.
- 4° Since the TIRS model is not a pure boolean query system, the counterintuitive results that can be obtained from such a system cannot occur.

Another list of advantages is obtained from Salton, Fox, and Wu's list of advantages for the p -norm model. It was stated earlier in this work that TIRS had all of the advantages of the p -norm model without the disadvantages. These advantages are

- 1° The p -norm model contains the boolean model.
- 2° The p -norm model contains the vector space model.
- 3° The p -norm model contains the fuzzy set theory model.
- 4° The choice of p -norm changes the query interpretation.
- 5° The p -norm system can be built on top of an inverted file system.

Of the advantages on this list, 4° was shown to be non-topological. That the remaining advantages are also advantages of the TIRS model will be shown later; cases 1° through 3° will be demonstrated in the next chapter, Model Unifications, and case 5° will be shown in the next section. In addition, it will be shown that certain other models can also be brought under the TIRS model, and that, because of the topological interpretation of the model, it may be possible to implement TIRS using faster data structures than are now being used. We begin by considering implementations of the TIRS model.

TIRS Implementations

The straight forward method of implementing TIRS is by means of an inverted file system. The requirements of such a system are well known, and construction of these systems is almost routine. Construction of a TIRS model as an inverted file system would require nothing new. Associate with each attribute term a set of pointers into the data, with one pointer for each document which has a non-zero value at that attribute. Sort this set by attribute value. We will call this sorted set a *projection*, using the standard terminology. Note that each projection is really nothing more than a portion of an inverted file, so the standard methods of storage will work. Document location is then a series of binary searches into the projections. Assuming that the average number of elements in each projection is n , and that the number of non-zero terms in the document to be located is k , then it takes time

$$O(k \log n) \tag{40}$$

to find a particular document, since each of the k inverted files must be searched using a binary search. This is of course the same time required by a regular inverted file system.

The process of query lookup is not much different from document location for the TIRS model. The major difference is that an interval is looked up at each point, instead of simply one point in the projection. But, since the projections are sorted, a range lookup only takes one point lookup and one scan. Hence, in each coordinate, it takes time

$$O(\log n) \tag{41}$$

to find the set of documents which are in the coordinate neighborhood of the query point, and, if there are l documents in that range, time

$$O(l) \quad (42)$$

to scan them. The total time is then

$$O(\log n + l) \quad (43)$$

per coordinate. If there is no additional processing time required to find the total distance of a point from the query point, given the coordinate distances, then the total query processing time is no worse than

$$O(k \log n + k \max_k \{l_k\}), \quad (44)$$

where l_k is the number of documents in the k th range.

In the discussion of the TIRS model, nothing has been said about the choice of metric to be used. It is clear that a metric that is easy to calculate is desired, but we see from the last paragraph that a desirable property for our metric is that it be easy to calculate from its projection distances. These two properties suggest that, of all the common metrics available, the p_∞ is probably the best, since

$$\text{dist}_\infty(D, Q) = \max_k \{ |d_k - q_k| \}_{k=1}^\infty \quad (45)$$

i.e., the distance is simply the largest of the (finite number of) non-zero coordinate differences. With this metric we obtain the processing time of the preceding paragraph.

Actually, we can do a little better, at least from a heuristic point of view. Note that, using the p_∞ metric, all that we need to do is to process the query in a coordinate-wise fashion. A point is never added to the list of points when a new coordinate is

examined, but can only be removed from the list of currently acceptable points, since once a coordinate difference in a point exceeds the allowed radius, the point is outside of the relevance ball. As a result, it makes sense to process the smallest query first, since this will reduce the processing time to

$$O(k \log n + k \min_k \{l_k\}). \quad (46)$$

Of course, we cannot be assured of finding the coordinate with the smallest number of points in it until we have tried all of the coordinates, but, in general, it makes sense to choose first the coordinate in which we expect the smallest number of points. [27]

It may turn out that this method of storage, while based on inverted files and hence capable of being added to existing systems, may not be the best one available for the TIRS. Several groups, including Bentley et al. [Ben75, Ben79a, Ben79b, Fri77], Flajolet and Puech [Fla86], Lloyd and Ramamohanarao [Lio82], and Nievergelt, Hinterberger, and Sevcik [Nie84] have been concerned with the problem of multidimensional retrieval. The authors of each paper above have described at least one method of retrieval which is faster than the inverted file case. Of the methods listed, Bentley's kd-trees seem to work the best for the TIRS system, giving a preprocessing time of

$$O(n \log n), \quad (47)$$

a storage requirement of

$$O(kn), \quad (48)$$

[27] Under the product space interpretation of the TIRS model, we can choose first the coordinate having smallest size.

and an average case query response time of

$$O(\log n + l). \quad (49)$$

The only disadvantage seems to be that the system would need to go offline occasionally in order to update and re-balance the data structure; this offline time can, however, be planned for.

Previous Work

Aside from the general similarity of TIRS to any vector space system, there is one system that TIRS is most like. This is the Generalized Vector Space Model (GVSM) described by Wong and Ziarko [Won85a, Won85b, Won85c]. In this section, the Generalized Vector Space Model will be described and contrasted with TIRS.

A GVSM is constructed by letting A be a set of attribute terms, and then constructing the set \bar{A} by letting

$$\bar{A} = 2^A, \quad (50)$$

i.e., \bar{A} is the set of subsets of A . An element of \bar{A} is to be understood as the logical conjunction of the terms in its subset. This allows for the representation of term interrelations and term dependencies by specification of the particular conjunction of terms which are to be considered the same. The space is then composed of coordinates from the set \bar{A} ; each coordinate has as its space the interval $[0,1]$. Documents and queries are placed in the space in the regular fashion, with the term dependencies now being taken into account.

As can be seen from the description just given, the Generalized Vector Space Model is essentially a vector space model, with both documents and queries

represented as points in a product metric space whose coordinates are indexed by attribute terms. Thus, it is already within the topological paradigm. The major difference between the GVSM and TIRS is that, in GVSM, the coordinates are logical conjunctions of attribute terms, rather than the terms themselves. This was done in an attempt to allow for term dependencies and interrelations; however, instead of simplifying matters it seems to have complicated them unnecessarily. TIRS can easily handle the problem of dependent terms through the construction of *quotient spaces* of the original space, as follows: For each collection of attribute terms that are related, measure the amount of similarity, and then mod out by the resulting relation. The resulting space is equivalent to one obtained by assigning weights of the appropriate amount to all of the related terms whenever one of the terms is used. It has the additional advantage of allowing term interrelationships of other than unity, which does not seem possible with a GVSM.

In terms of the ultimate aim of allowing representation of term dependencies, either method will work. The question that should be asked now is, "Which method for the resolution of term dependencies is better?" Since either system will achieve the desired results, a choice must be made on other considerations, such as simplicity, practicality, and utility of the resulting user interface. It would seem that the TIRS model is both simpler and more practical than the GVSM model. The major problem encountered in GVSM is the size of the resulting space. In TIRS, the space is assumed to be countably infinite in dimension, thus allowing for an arbitrarily large number of attributes. In practice, of course, the dimension of the space is the size of the underlying attribute set, and, as each new attribute is added, the dimension of the

space increases only by one. In the GVSM system, on the other hand, the space theoretically has the same cardinality as the reals, i.e., not countable. There is no way to represent this many items in any known data structure. In practice, this lack is not a problem, since only a finite number of attribute terms are actually used, causing the resulting space to have finite dimension. However, as each new attribute term is added, the dimension of the resulting space doubles. Indeed, it appears that the amount of work necessary to answer a query actually is increased by a factor of four with each new attribute term, since a query is handled by a matrix computation having dimension the dimension of the space. It is expected that most information retrieval systems will have constantly changing contents. If one uses automatic term indexing, as most systems are expected to do, then any such dynamic system must be able to easily handle the addition of new attribute terms. It seems that a GVSM model cannot do so as simply as can a TIRS model.

A second problem with a GVSM model is that its queries are necessarily limited. Since the space in the GVSM model already has built into it the idea of term dependencies as logical conjunctions, it is going to be difficult to allow the use of boolean queries in any reasonable manner. It would seem that it is better to keep the terms separate and allow boolean combinations to occur later, in the queries, than to use the boolean operators in the construction and not allow the use of boolean queries.

Finally, it seems that there is at least a potential problem with maintenance of the system's internal consistency. As an example, consider the situation where the terms a_1, a_2 , and a_3 have been determined to be dependent. Let us further assume, as a simplification, that the terms are synonyms, so that there is a weight of one to be

assigned between the three pairs of elements. In order for the system to be consistent, this three term dependency requires that the three pairs

$$\{a_1, a_2\} \quad (51)$$

$$\{a_1, a_3\} \quad (52)$$

$$\{a_2, a_3\} \quad (53)$$

all have a term dependency weight of one, as well as the original triple

$$\{a_1, a_2, a_3\}, \quad (54)$$

and that all new documents which are entered into the system having a non-zero weight in *any* of the seven distinct coordinates involving any of the a_i have *all* of these seven coordinates updated consistently. Not only does this require exponentially more work than simple one coordinate entry, but it leaves open the possibility that data may be entered differently in different places. The prevention of internal inconsistency in database systems is a problem with known solutions; the standard solution is to store the data non-redundantly [28], thus destroying the GVSM. Note that requiring that data be stored non-redundantly is essentially nothing more than passing to the quotient space in TIRS.

As can be seen from the above discussion, it seems that the Generalized Vector Space Model suffers from a design flaw similar to that in the p -norm model: in the attempt to clean up known problems of earlier models, the structure is made more complex, rather than less complex. While this type of construction will not invariably lead to problems, it certainly has the potential to do so. This potential seems to have been achieved in the case of the GVSM.

[28] See, e.g., Ullman [Ull82], Chapter Ten.

Summary

The TIRS model and the topological paradigm have now been described. In particular, it was shown that the TIRS system avoids the problems inherent in both the p -norm model and GVSM while retaining all of their claimed advantages, and in addition allows the possibility of a much smaller run time for query processing. Only one thing remains to be done. It has been claimed that all of the standard models of information retrieval systems live within the topological paradigm. The proof of this statement is the subject of the next chapter; for each model, it will be shown that each of the four axioms of the paradigm is satisfied.

Chapter Four

Model Unifications

The demonstration that the topological paradigm is sufficiently general to include the models mentioned earlier is the subject of this chapter. Each model will have its own demonstration, beginning with the boolean model. This will be followed by the p -norm model, the fuzzy set theory model, and the hierarchical model.

The Boolean Model

In the boolean model, a document is considered to be a collection of attribute terms, and a query is essentially a boolean combination of attribute terms. In order to cast this model into the topological paradigm, all that is necessary is to let our document space be the space

$$\prod_{i \in A} \{0, 1\}_i, \quad (55)$$

where A is the set of all possible attribute terms. Under this interpretation, the required distance function is simply the identity metric, i.e.,

$$\text{dist}(x, y) = \begin{cases} 0 & \text{if } x = y \\ \infty & \text{otherwise} \end{cases} \quad (56)$$

which is easily shown to be a distance function: the function is non-negative, is zero if and only if the points are the same, is symmetric, and satisfies the triangle inequality (if the points are different, one of the distances is infinite). Since any non-contradiction query can be placed into disjunctive normal form, a query can be represented as a finite set of points in the document space using the same process as for the TIRS model.

From the above description, it can be seen that the four properties of the topological paradigm hold; the space is a product space (property two) of weighted (property three) attribute terms (property two) which is a metric space (property one), and queries are nothing more than points in this space (property four). Thus the boolean model can be considered as an instance of the topological paradigm.

The p -Norm Model

Since the TIRS model is built on the p -norm model, it should be easy to fit the p -norm model into the topological paradigm. Several of the the required properties of the topological paradigm come from the p -norm model: the p -norm itself is defined only on a metric space, while the underlying space is a product space of attribute weights. The only property that needs to be derived is property four: that queries should be interpretable as points in the document space. It is easy to place a query into the document space based on its term weights; the only question is what happens to the resulting space. Since there are similarity functions defined on the space, they must somehow be accounted for. Since the similarity functions are not distances in the general sense, but rather distances from the origin and identity, it is not possible for the similarity measures to be metrics on the space. In particular, note that

$$\text{sim}(A, B) \neq \text{sim}(B, A), \quad (57)$$

where A and B are points in the document space, with the second being interpreted as a query in each case. Also, note that for any point A (whether query or document) in DS we have

$$\text{sim}(A, A) \neq 0. \quad (58)$$

Since these two properties are two of the axioms that any metric must obey, it is impossible for the similarity measures to be metrics.

If these functions cannot be metrics, it should still be possible to interpret the measures as something useful, or the paradigm would not be a very useful tool. Perhaps the best way to interpret the result of considering queries as points in the space is to realize that each of the similarity measures gives rise to a topological semigroup [29], with the underlying topology being that of the document space topology (with included query points), and the continuous binary operation being just the similarity measure itself, extended coordinatewise. Salton, Fox, and Wu show that the fuzzy set theoretic model can be considered as a case of the p -norm model; under this interpretation it is seen that the p -norm model can be considered as a case of the fuzzy set theoretic model, since a fuzzy set theoretic model is simply a pair of topological semigroups defined using the standard max and min operators [30]. This interpretation seems to be a natural one, so that putting the queries into the document space is probably acceptable.

There may have been some question during this demonstration as to why the placement of the p -norm model into the topological paradigm was not very easy, since the TIRS model is a direct outgrowth of the p -norm model. The answer to this question is that the TIRS model is perhaps what the p -norm model would have been, had one been trying to begin with the topological paradigm and design a model based on

[29] For an introduction to topological semigroups, consult the works by Carruth, Hildebrandt, and Koch [Car83, Car86].

[30] This result should not be too surprising, since both models are actually just instances of the topological paradigm.

standard distances. The idea of distance between points is certainly a part of the p -norm model, but the idea is lost when one commits to measuring distance only from two places. To fix this problem easily requires something like the TIRS model; however, even without it, the p -norm model still fits within the topological paradigm.

The Fuzzy Set Theoretic Model

In one sense, this demonstration is already finished, since Salton, Fox, and Wu have shown that the fuzzy set theoretic model is an instance of the p -norm model. However, it is possible to show directly that the fuzzy set theoretic model is an instance of the topological paradigm, and this direct demonstration is much simpler than in the p -norm case. Hence, the direct demonstration will be given.

As before, the fuzzy set theoretic model easily fulfills the first three properties of the topological paradigm; the only property that is not immediately apparent is the requirement that queries fit into the document space. This can be done by having Q_A and/or B represented as a pair of points, one at the one position in the A direction, and the other in the B direction, also of value one [31]. The fuzzy query is then evaluated by having

$$\text{sim}(D, Q) = \begin{cases} \max_{i \in T} \{ \text{dist}(d_i, q_i) \} & \text{or operator} \\ \min_{i \in T} \{ \text{dist}(d_i, q_i) \} & \text{and operator} \end{cases} \quad (59)$$

i.e., just measuring coordinatewise distance in the space, and applying one of two

[31] Actually, it does not seem to be necessary to require that the query be placed at the one position of each coordinate, only that a point having exactly one non-zero coordinate be placed in the space for each term in the query. This, if true, would allow weighted fuzzy queries.

standard topological semigroup operators to the result. Since the queries can be so easily placed into the document space, the fuzzy set theoretic model must be an example of the topological paradigm.

The Probability Model

In the original description of the probabilistic model [Mar60], an attempt is made to place the description on a topological footing. In particular, the collection of documents is described as a document space, and an attempt is made to describe a distance in this space [32]. Since a document is defined by its index terms in this model, the document is a product space, with the elements of the probability matrix serving as term weights. Once again the first three properties of the topological paradigm are seen to hold for the model; in this case, however, Maron and Kuhns went on to define another space for the queries, one which had many of the same topological properties of the document space. Since there are two distinct spaces involved in the probabilistic model, it would seem difficult to place this model in the topological paradigm, even though three of the four properties are met.

However, this difficulty is more apparent than real. Maron and Kuhns feel that the document space should be different from the query (request) space, since one has a true metric in the document space but only a measure of "closeness" [33] in the query space. The reasoning seems to be, that, since there are different topological properties involved in the two spaces, the spaces must be distinct. But note that none

[32] [Mar60], p. 224.

[33] [Mar60], p. 224.

of the semantic and statistic relationships described in the request space, when considered topologically, prohibit the introduction of a metric on the space. Indeed, the "coefficients of association" [34] have the essential property of distances, in that whenever there are coefficients of association which hold between all pairs of a three element set, then the triangle inequality holds. The other properties of a metric can be obtained by requiring that the distance from a point to itself be zero, and by defining the distance between pairs of points which are not associated to be infinity. Maron and Kuhns seem to realize this, for they later convert the coefficients of association into a distance [35], which allows property four of the paradigm to be satisfied by taking the union of the two spaces. The metrics must be the same in the two spaces, since they both arise from the product of intervals, and the space is finite dimensional. Thus Maron and Kuhns have themselves placed the probabilistic model into the topological paradigm.

The preceding description has been based on an early paper, on work that has been followed by many developments. The "standard" model of probabilistic information retrieval is based on work by Robertson, Maron, and Cooper [Rob82]. In this paper, the work described above is only one model of four that are available. However, the ideas of document space and query space carry over to the other three models described here. In fact, the most general model, Model 3, is really nothing more than a product space derived from the spaces of Models 1 and 2, where the product topology is defined, in the usual manner, from the topologies of the coordinate

[34] [Mar60], p. 224, footnote.

[35] [Mar60], page 229.

spaces. Since model 2 can be topologized (with query points placed in the document space) using the same method as described above, and since model 0 is a quotient space of model 3, it follows that all of the current probability models of information retrieval fall within the topological paradigm.

The Hierarchical Model

Each of the models discussed previously have been a part of the mainstream of research in information retrieval. As an example of a model in which properties one through three of the topological paradigm are difficult to confirm, but property four is easy, the hierarchical model of Kolodner [Kol83] is considered. This model, which is a typical example of the artificial intelligence approach to information retrieval, is based on a hierarchical arrangement of information. Attributes are arranged in a tree structure; a parent is considered to be a more inclusive attribute than its child. In this model, it is desired to find whether a "fact" is in the system, rather than a document about something, but note that a fact and a document are both described, for our purposes, by the attributes associated with them. In the hierarchical model, one finds a fact by locating where the query would be; if there is a fact at that location, then it must be the correct one. Thus property four of the topological paradigm is satisfied.

To show that the other three properties of the paradigm are satisfied, we must construct a weighted product space having a metric as the basis of its topology. The metric can be derived from the weights, if an appropriate set is chosen. One such set is the set

$$B = \{0, 1\}, \quad (60)$$

using the real metric. The space to be constructed is then the product

$$\prod_{i=1}^{\infty} B, \quad (61)$$

with the product topology. All that remains is to fit the hierarchical scheme into this space, in a fashion that preserves the ability to create new subcategories as needed. This is done by mapping the nodes in the hierarchy to coordinates in the space by means of the following function f . First, give each node a sequence number, based on its position in the hierarchy, as follows:

- 1° The sequence number of the root is 1.
- 2° If the sequence number of a node is x , then the sequence number of x 's i th child is x, i .

Thus, the second child of the third child of the root would have sequence number 1,3,2. The second part of the map from the hierarchy to the space is to associate each sequence number with a particular coordinate. Any such map will work, provided that it is one-to-one. One such map that is particularly appealing in this situation is the map which considers the sequence as a number in base eleven, with the comma serving as the eleventh symbol, and maps into the coordinate of the same number. It is clear that this map is one-to-one, so it will serve as an adequate conversion from the hierarchy into the space. Hence, all five of the topological paradigm properties hold, and this model, along with the earlier ones, is simply one more example of the topological paradigm.

Summary

In this chapter, it was shown that each of the standard models of information retrieval could be considered as cases of the topological paradigm. This construction is more than simply a theoretical result; it provides a basis for the following statement: Since all of the models of information retrieval can be explained using the same system, then this system should be considered the "natural" one. As such, any specific example of this system, such as the TIRS model, should be investigated closely.

Chapter Five

Conclusion

There were three main topics discussed in this work. The first topic was an investigation of the p -norm model of Salton, Fox, and Wu. In this section, three things were shown.

- 1° The use of p -norms to indicate levels of meaning intermediate between logical “and” and logical “or” causes no change in the topology of the underlying document space.
- 2° There exist distinct documents in the p -norm model that cannot be separated by any choice of p -norm (for fixed query weights). With varying query weights and fixed values of p , the separation problem disappears.
- 3° The use of weights in the queries causes no change in the topological structure of the document space.

In light of these three results, the utility of the p -norm similarity measure is questionable, since the use of weighted queries seems to have just as much power without the additional computational overhead. In particular, if there is a system that can use the weighted aspects effectively, then there would be little use for the p -norm model.

In the next chapter, the topological paradigm was defined. The major justification for this paradigm is the sense that the only really important idea in most information retrieval models is the idea of distance between either pairs of documents or between a query and a document. The paradigm is then simply the minimal set of requirements necessary to guarantee good behavior of the metric space. As an exam-

ple of the utility of the topological paradigm, the TIRS model was constructed. The main features of the TIRS model include the use of a countably infinite dimensioned topological space, and a computationally simple distance function.

There are four types of queries that can be asked of the TIRS system, ranging from a natural language vector space type query to a query that can fully utilize the particular features of the model. The second query type, the weighted boolean queries, is important to consider, as this model provides the first recorded instance of weighted boolean queries which obey the standard lattice laws at the atomic level. This feature was proved through the construction of a set homomorphism from the queries onto all finite combinations of points in the document space. Finally, in this chapter it is shown that the TIRS model retains all of the claimed features of the p -norm model, while having the advantages of conceptual simplicity and multiple means of implementation, several of which appear to be asymptotically faster than any other model available. Hence, the TIRS model should be easy to build and should run extremely rapidly once built.

In the last chapter, the topological paradigm was once again considered. Five models of information retrieval, the boolean model, the vector space model, the fuzzy set theoretic model, the probabilistic model and the hierarchical model, were shown to be specific instances of the topological paradigm. In no case was the construction especially onerous; the most difficult unification occurred with the p -norm model, which is understandable since a basic aim of the topological paradigm is to prevent some of the problems which occur in the p -norm model from happening. The importance of this work is as a justification of the topological paradigm itself; if the major

models of information retrieval are *all* instances of the topological paradigm, then one must strongly suspect that a central concept of the field has been identified.

Future Work

There is much work left to do concerning the topological paradigm and the TIRS model, ranging from the very practical to the very theoretical. Several of the main jobs that remain are listed below, in no particular order.

- 1° More work needs to be done with the perceived problems in the p -norm and the GVSM models. The topological nature of these models should be more fully explicated.
- 2° The TIRS model should be constructed, with an eye to commercial applications. This would be a multistage project, consisting of at least the following steps: A test version of TIRS should be constructed, in order to determine which of the five or six possible implementations is the best. Work should be done on automating the indexing and weighting of documents. Much work has already been done in this area; discovering which of the possibilities works best for TIRS is needed. A medium sized (1000 - 10000 documents) TIRS should be constructed, and compared with systems such as SMART and SIRE. Using this system, a comparison of the different query methods should be made. Which of the last three types of queries mentioned is the best for general use? This question needs to be answered from both a user oriented and theory oriented view; the answers could easily be different. Assuming that all goes well in the earlier stages, then a prototype commercial version of TIRS should be built.

- 3* Many systems rank documents today. A comparison of these rankings should be done; since the TIRS system has a particularly simple means of document ordering, the hope exists that the resulting rankings should be better than ones obtained from a more complex procedure.
- 4* Even the best systems today do not perform too well. Many investigators, such as Dillon, Ulmschneider, and Desper [Dil83], are switching to some method of relevance feedback as an aid to document retrieval. The distances defined in the TIRS model and the ability to use finite sets of perfect points allow the use of many methods of feedback. In addition, arbitrarily complex queries can be constructed in a natural manner, through the use of set partitionings via relevance radii cutoffs.
- 5* The unifications should be tightened and extended. Although the major models have been shown to be instances of the topological paradigm, many other retrieval models exist. It should be possible to unify most of these models, also.

As can be seen from the above list, there is still work left to be done. If everything works, then the topological paradigm must be seen as the proper archetype of information retrieval. Even if not everything can be made to fit, however, the paradigm has some utility, and has already given rise to a system which is both simple and fast.

Bibliography

- [Bar85] Bartschi, M., "Requirements for query evaluation in weighted information retrieval," *Information Processing and Management*, vol. 21, no. 4, pp. 291 - 303, 1985.
- [Ben75] Bentley, J. L., "Multidimensional binary search trees used for associative searching," *CACM*, vol. 18, no. 9, pp. 509 - 517, 1975.
- [Ben79a] Bentley, J. L., "Multidimensional binary search trees in database applications," *IEEE Trans. on Softw. Eng.*, vol. SE-5, no. 4, pp. 333 - 340, July 1979.
- [Ben79b] Bentley, J. L. and Friedman, J. H., "Data structures for range searching," *Computing Surveys*, vol. 11, no. 4, pp. 397 - 409, Dec 1979.
- [Boo74] Bookstein, A. and Swanson, D. R., "Probabilistic models for automatic indexing," *J. ASIS*, vol. 25, no. 5, pp. 312 - 318, Sept - Oct 1974.
- [Bue81a] Buell, D. A., "A general model of query processing in information systems," *Information Processing and Management*, vol. 17, no. 5, pp. 249 - 262, 1981.
- [Bue82] Buell, D. A., "An analysis of some fuzzy subset applications to information retrieval systems," *Fuzzy Sets and Systems*, vol. 7, pp. 35 - 42, 1982.
- [Bue81b] Buell, D. A. and Kraft, D. H., "Threshold values and Boolean retrieval systems," *Information Processing and Management*, vol. 17, pp. 127 -

136, 1981.

- [Bue81c] Buell, D. A. and Kraft, D. H., "A model for a weighted retrieval system," *J. ASIS*, vol. 32, no. 3, pp. 211 - 216, May 1981.
- [Car83] Carruth, J. H., Hildebrandt, J. A., and Koch, R. J., *Theory of Topological Semigroups*, vol. 75, Pure and Applied Mathematics, Marcel Dekker, New York, 1983.
- [Car86] Carruth, J. H., Hildebrandt, J. A., and Koch, R. J., *Theory of Topological Semigroups II*, vol. 100, Pure and Applied Mathematics, Marcel Dekker, New York, 1986.
- [Cat84] Cater, Steven C, Iyengar, S. Sitharama, and Fuller, John, "Computation of logical effort in high level languages," *Computer Languages*, vol. 9, no. 3/4, pp. 133 - 148, Dec 1984.
- [Coo82] Cooper, W. S. and Huizinga, P., "The maximum entropy principle and its application to the design of probabilistic retrieval systems," *Information Technology: Research and Development*, vol. 1, pp. 99 - 112, 1982.
- [Dil83] Dillon, M., Uimschneider, J., and Desper, J., "A prevalence formula for automatic relevance feedback in boolean systems," *Information Processing and Management*, vol. 19, no. 1, pp. 27 - 36, 1983.
- [Dug66] Dugundji, J., *Topology*, Allyn and Bacon, Inc., Boston, 1966.
- [Flaj86] Flajolet, P. and Puech, C., "Partial match retrieval of multidimensional data," *JACM*, vol. 33, no. 2, pp. 371 - 407, April 1986.

- [Fri77] Friedman, J. H., Bentley, J. L., and Finkel, R. A., "An algorithm for finding best matches in logarithmic expected time," *ACM Trans. Math. Softw.*, vol. 3, no. 3, pp. 209 - 226, Sept 1977.
- [Kol83] Kolodner, J. L., "Indexing and retrieval strategies for natural language fact retrieval," *ACM Transactions on Database Systems*, vol. 8, no. 3, pp. 434 - 464, Sept 1983.
- [Kra85] Kraft, D. H., "Advances in information science: where is that record?," *Advances in Computers*, vol. 24, pp. 277 - 318, 1985.
- [Kra78] Kraft, D. H. and Bookstein, A., "Evaluation of information retrieval systems: a decision theory approach," *J. ASIS*, pp. 31 - 40, Jan 1978.
- [Kra83] Kraft, D. H. and Buell, D. A., "Fuzzy sets and generalized boolean information retrieval," *International Journal of Man-Machine Studies*, vol. 19, pp. 45 - 56, 1983.
- [Lan70] Lang, Serge, *Algebra*, Addison Wesley, Reading, MA, 1970.
- [Llo82] Lloyd, J. W. and Ramamohanarao, K., "Partial-match retrieval for dynamic files," *BIT*, vol. 22, pp. 150 - 168, 1982.
- [Mar60] Maron, M. E. and Kuhns, J. L., "On relevance, probabilistic indexing and information retrieval," *JACM*, vol. 7, pp. 216 - 244, July 1960.
- [Nie84] Nievergelt, J., Hinterberger, H., and Sevcik, K. C., "The grid file: An adaptable, symmetric multikey file structure," *ACM Trans. Database Syst.*, vol. 9, no. 1, pp. 38 - 71, Mar 1984.

- [Rad83a] Radecki, T., "Generalized boolean methods of information retrieval," *Int. J. Man-Machine Studies*, vol. 18, no. 6, pp. 407 - 439, 1983.
- [Rad83b] Radecki, T., "A theoretical background for applying fuzzy set theory in information retrieval," *Fuzzy Sets and Systems*, vol. 10, pp. 169 - 183, 1983.
- [Rad84] Radecki, T., "Foundations of fuzzy information retrieval," Technical Report TR84-019, Louisiana State University Department of Computer Science, 1984.
- [Rob82] Robertson, S. E., Maron, M. E., and Cooper, W. S., "Probability of relevance: a unification of two competing models for document retrieval," *Information Technology: Research and Development*, vol. 1, pp. 1 - 21, 1982.
- [Sal83a] Salton, G., Buckley, C., and Fox, E. A., "Automatic query formulations in information retrieval," *J. ASIS*, vol. 34, no. 4, pp. 262 - 280, July 1983.
- [Sal83b] Salton, G., Fox, E. A., and Wu, H., "Extended boolean information retrieval," *CACM*, pp. 1022 - 1036, 1983.
- [Sal83c] Salton, G. and McGill, M. J., *Introduction to Modern Information Retrieval*, McGraw-Hill, New York, 1983.
- [Ull82] Ullman, J. D., *Principles of Database Systems*, Computer Science Press, Rockville, MD, 1982.
- [Wal79] Waller, W. G. and Kraft, D. H., "A mathematical model of a weighted Boolean retrieval system," *Information Processing and Management*,

vol. 15, no. 5, pp. 235 - 245, 1979.

- [Won85a] Wong, S. K. M. and Ziarko, Wojciech, "A Unified Model in Information Retrieval II," Technical Report CS-85-08, University of Regina Computer Science Department, May, 1985.
- [Won85b] Wong, S. K. M. and Ziarko, Wojciech, "Query Processing in Information Retrieval Systems Based on Generalized Vector Space Model," *Proceedings of the Fourth Hungarian Computer Science Conference*, July, 1985.
- [Won85c] Wong, S. K. M., Ziarko, Wojciech, and Wong, P. C. N., "Generalized Vector Space Model in Information Retrieval," *Proceedings of the ACM-SIGIR Conference on Research and Development in Information Retrieval*, Montreal, Canada, June, 1985.

Appendix

A Review of Algebra and Topology

In this appendix are the topological and algebraic results needed to understand the work in the earlier chapters. The section is divided into two parts; the first part is a survey of algebra, and the second part is a survey of topology. More information is available on these topics; proofs of most of the theorems mentioned in this section can be found in Lang, [Lan70], for the algebraic topics, and in Dugundji, [Dug66], for the topological parts. For the specific topic of topological semigroups, see Carruth, Hildebrandt, and Koch. [Car83, Car86].

Preliminaries

In this section definitions are given which apply to both the algebraic and topological discussions appearing later.

The term *set* is an undefined term. It is normally considered to be a collection of items, which may themselves be sets. The basic property of a set is *membership*;

$$a \in A \tag{62}$$

means that a is a *member* (or *element*) of the set A . In this case a may or may not be a set. A set A is a *subset* of a set B ,

$$A \subseteq B, \tag{63}$$

if, for each element $a \in A$ it is also the case that $a \in B$, i.e.,

$$A \subseteq B \stackrel{\text{def}}{=} \forall x, x \in A \Rightarrow x \in B. \tag{64}$$

Two sets A and B are *equal* if each contains the other, or

$$A = B \stackrel{\text{def}}{=} A \subseteq B \wedge B \subseteq A. \tag{65}$$

There are certain special sets that deserve mention. The *empty set*, \emptyset , is a set containing no members. Most axiomizations of set theory assume that there is only one empty set. The *universal set*, or *domain of discourse*, commonly denoted U , is the set which contains everything currently under discussion. For technical reasons it is not possible to have only one universal set; it must be restricted to an appropriate domain. An *ordered pair* (a, b) is a set of the form

$$\left\{ \{a\}, \{a, b\} \right\} \quad (66)$$

Ordered pairs are used when it is necessary to distinguish the position of the elements in a two element set. *Ordered n -tuples* are defined recursively:

- 1° An *ordered triple*, or *ordered 3-tuple*, is an ordered pair whose first element is an ordered pair, and whose second element is a singleton.
- 2° An *ordered n -tuple* is an ordered pair whose first element is an ordered $n-1$ tuple, and whose second element is a singleton.

Operations on sets include intersection, union, set difference, and Cartesian product. These operations are defined as:

Let $A = \{a_i\}$ and $B = \{b_i\}$ be sets. Then

$$A \cup B = \{c \in U \mid c \in A \text{ or } c \in B\} \quad (67)$$

$$A \cap B = \{c \in U \mid c \in A \text{ and } c \in B\} \quad (68)$$

$$A \times B = \{(a, b) \mid a \in A, b \in B\} \quad (69)$$

Let A and B be sets. A *relation* (specifically, a *binary relation*) ρ from A to B is a subset of $A \times B$. A is the *domain* of the relation, and B is the *codomain* of the relation. There are several special types of relations. Let $\rho \subseteq A \times B$. The *inverse rela-*

tion ρ^{-1} is a relation from B to A defined as

$$\rho^{-1} =_{df} \{ (b, a) \mid a \in A, b \in B \}. \quad (70)$$

A relation in which no element of the domain appears in more than one ordered pair (as the first element) is a *partial function*. A partial function in which each element of the domain occurs as the first element of an ordered pair is a *function*, or *total function*, if one wishes to emphasize that all of the domain elements occur. The notation for a function f with domain A and codomain B is $f: A \rightarrow B$. A function $f: A \rightarrow B$ is *one-to-one* if different elements map to different values, i.e.,

$$\forall x, y \in A, x \neq y \Rightarrow f(x) \neq f(y) \quad (71)$$

A function $f: A \rightarrow B$ is *onto* if

$$\forall b \in B, (\exists a \in A) f(a) = b \quad (72)$$

that is, if each element in the codomain has a preimage. A function which is both one-to-one and onto is a *bijection*. A function $f: A \times A \rightarrow A$ is called a *binary operation*.

Properties of Relations

There are several properties that a relation can possess. A relation $\rho \subseteq A \times A$ is *reflexive* if

$$\forall a \in A, (a, a) \in \rho. \quad (73)$$

ρ is *symmetric* if

$$(a, b) \in \rho \Rightarrow (b, a) \in \rho. \quad (74)$$

ρ is *transitive* if

$$(a, b) \in \rho \wedge (b, c) \in \rho \Rightarrow (a, c) \in \rho. \quad (75)$$

A relation which is reflexive, symmetric, and transitive is an *equivalence* relation.

Algebra

The basic structure in all of algebra is that of a set with one or more binary operations. The easiest structures to consider are those which have only one operation defined on them; these structures are the ones that will be considered here.

There are several properties that a binary operation can have. Some of the more important ones are the following. An operation $\cdot : A \times A \rightarrow A$ (the function is normally written $a \cdot b$) is *associative* if

$$\forall a, b, c \in A, (a \cdot b) \cdot c = a \cdot (b \cdot c). \quad (76)$$

An operation is *commutative* if

$$\forall a, b \in A, a \cdot b = b \cdot a. \quad (77)$$

An element $e \in A$ is an *identity* for A if

$$\forall a \in A, e \cdot a = a = a \cdot e. \quad (78)$$

If a structure has an identity e , then, given an element $a \in A$, an element $b \in A$ having the property

$$a \cdot b = e = b \cdot a \quad (79)$$

is called an *inverse element* for a . If each element in a structure has an inverse element, then the structure is said to *have inverses*. A set with a binary operation on it is called a *groupoid*. If the operation in a groupoid is associative, the structure is a *semigroup*. A semigroup with an identity element is a *monoid*. A monoid with inverses is called a *group*. If any of the operations on these structures is commutative, then the structure is *abelian*; hence, one can speak of an abelian semigroup or an abelian

group.

A basic question of algebra is, given two structures of the same type, say semigroups, are they really the same? This question is an important one, since the important part of an algebraic structure is the operation (or operations), and not the particular elements in the set that are being operated on. Surely one has the same structure, for all practical (algebraic) purposes, if one merely changes the names of the elements without changing the operation itself. As a result, the concept of *isomorphism* is utilized. In essence, two algebraic structures of the same type are isomorphic if they are really the same, except possibly for a consistent change of name of the elements. Formally, this is defined as follows.

Definition: Let A and B be structures of the same type. A and B are *isomorphic* if there exists a function $f: A \rightarrow B$ which is a bijection, and which preserves the defining properties of the type.

Consider the specific examples of set-isomorphism, semigroup-isomorphism, and abelian group-isomorphism. The defining property of a set is that it have elements. Hence, all that need be done to show that two sets are isomorphic is to find a bijection between them. As a result, two sets are isomorphic if and only if they have the same number of elements. This may seem a little odd, but remember that the essential properties in algebra are those on the operations, and that a set has no operations on it. Two semigroups are isomorphic if there is a bijection between them which preserves the defining properties of semigroups. Since the only defining property of a semigroup is an associative multiplication, we must answer two questions: does the operation send the same things to the same things, and does it preserve associativity? The

first part of the question can be answered by checking that

$$f(x \cdot y) = f(x) \cdot f(y) \quad (80)$$

for all x and y in A . Theoretically, the second part must be answered in a similar fashion, by checking that

$$f((x \cdot y) \cdot z) = f(x \cdot (y \cdot z)), \quad (81)$$

but note that an affirmative answer to the first question automatically answers the second question affirmatively. Hence, all that must be shown is the first part, which we will call *preserving the operation*.

A similar situation exists for abelian groups. The defining properties of abelian groups are commutativity, associativity, existence of identity, and existence of inverses. To show that the operation is preserved we must show that

$$f(x \cdot y) = f(x) \cdot f(y). \quad (82)$$

This guarantees that associativity is preserved. Similarly, if one of the two groups is commutative, then so must the other one, so commutativity is preserved. It is also possible to show that the image of the identity or an inverse acts like an identity or inverse; this, combined with the fact that identities and inverses are unique if they exist shows that these two properties are preserved. Once again it is the case that all that has to be shown is that operation is preserved; the rest comes for free.

It turns out that the idea of isomorphism is too strong an idea to be of real use in algebra, since there aren't really that many things that are exactly alike. The important part of each of the earlier examples was the idea of preserving the operation. It is this idea that we wish to keep, while removing other requirements. The only other properties of an isomorphism are that of being one-to-one and being onto. For

technical reasons the onto requirement is kept, and the one-to-one requirement is dropped. This generalization of isomorphism that results is a *homomorphism*, and, if $f: A \rightarrow B$ is a homomorphism, then B is said to be the *homomorphic image* of A . The major result of all of these definitions is the *first isomorphism theorem*. There is a first isomorphism theorem for each type mentioned above, but they all have the same proof, and so are essentially the same theorem. Let A and B be structures of the same type, and let B be the homomorphic image of A . We will define a new structure C , of the same type as A , as follows. An element of C is a set of elements of A , in particular, those elements that map to the same value under the homomorphism. The operation on C is defined similarly: if c_1 and c_2 are elements of C , then their product

$$c_1 \cdot c_2 \quad (83)$$

is defined to be the set containing the preimages of the unique value in B obtained by multiplying in A a value from c_1 and a value from c_2 . The first isomorphism theorem states that:

- 1° The structure C so obtained is of the same type as A and B ,
- 2° C is the homomorphic image of A , under a standard homomorphism, and
- 3° B and C are isomorphic.

The theorem further states that any time there is a structure with a *congruence relation* on it (essentially an equivalence relation which preserves the operation) that we may define a homomorphism in a fashion similar to the one defined from A to C , and then find an appropriate B . In essence, then, homomorphisms look like equivalence relations which preserve the operation, and vice versa.

Topology

In the field of general, or point-set, topology, the basic questions are those which attempt to discover the basic aspects of geometry and analysis. Some important concepts are connectedness, betweenness, and distance. Each of these ideas is a generalization of a concept in geometry; in addition, the idea of distance is central to all of analysis. Topology is the field which attempts to make sense of these ideas using as little structure as possible, in the same fashion as algebra attempts to make sense of the idea of sameness using minimal structure.

In this section, three areas will be covered: basic ideas, functions on spaces, and separation axioms. This is only a small part of topology, but it is all that is really needed for an understanding of the ideas in earlier chapters.

Definition: Let S be a set of objects, normally called *points*. The ordered pair (S, τ) is a *topology* or *topological space*, where τ is a collection of subsets of S , subject to

- 1° The universal set S and the empty set \emptyset are members of τ .
- 2° If $\{A_i\}_i$ are members of τ , then so is

$$\bigcup_i A_i \quad (84)$$

i.e., *arbitrary* unions of τ are in τ .

- 3° If A_1, A_2, \dots, A_n are members of τ , then so is

$$\bigcap_{i=1}^n A_i \quad (85)$$

i.e., *finite* intersections of τ are in τ .

The elements of τ are called the *open sets* of the topology. Note that changing τ changes the topological structure; in general, however, τ is not referred to explicitly, and S alone is said to be the topological space.

Let x be a point in a topological space (X, τ) . If $t \in \tau$ is such that $x \in t$, then t is said to be a *neighborhood* of x .

Other Ways to Define a Topology

There are several other ways that a topology can be defined. Three of these are through the use of *closed sets*, through a *basis* for the topology, or through a *subbasis* for the topology. Each of these will be considered.

If, in the definition of topological space given above, one interchanges the union and intersection operators, then one has an equivalent definition for topology based on *closed sets*, rather than the open sets previously defined. Note that if (X, τ) is a topology based on open sets, then an equivalent topology (X, τ') based on closed sets can be obtained by defining τ' as

$$\tau' = \{X \setminus t \mid t \in \tau\}, \quad (86)$$

hence, the two definitions are equivalent.

It is not always necessary to fully specify τ in order to obtain a topology. Let X be a set of points, and let K be a collection of subsets of X . A topological space (X, τ) can be constructed from K through the following construction:

- 1° Let $K_1 = K \cup \emptyset \cup X$.
- 2° Let K_2 consist of all finite intersections of elements of K_1 .

3^{*} Let τ consist of unions of arbitrarily many elements from K_2 .

Clearly τ fulfills the requirements of a topology, since it is constructed by following the algorithm implicit in the definition. The original set K is called a *subbasis* for the topology. Note that any set can serve as a subbasis for a topology.

It is also possible to construct a topology by taking arbitrary unions of a set, provided that the set already has the finite intersection property. Such a set is said to be a *basis* for the topology. Clearly, not every set can serve as a basis for a given topology.

Functions on Topological Spaces

The basic property of functions which have a topological space as either domain or codomain is that of *continuity*. Let $f: X \rightarrow Y$, where (X, τ) and (Y, σ) are topological spaces. f is *continuous* if the relation f^{-1} takes open sets to open sets, i.e., if

$$S \in \sigma \Rightarrow f^{-1}(S) \in \tau. \quad (87)$$

Note that f^{-1} , rather than f , is chosen since f^{-1} preserves union, intersection, and set difference, while f does not always preserve intersection and set difference. The importance of a function's being continuous is that the topological properties are preserved from the codomain to the domain. Since a topology is defined only in terms of its set of points and its open sets, a function that maps open sets to open sets (f^{-1}) should preserve most of the properties which derive from the open sets.

There are five cases of special importance.

Definition: Let S be a topological space, and define a function $d: S \times S \rightarrow [0, \infty]$. d is said to be a *metric*, (and S a *metric space*) if

- 1° $d(x, x) = 0$.
- 2° If $x \neq y$, then $d(x, y) > 0$.
- 3° $d(x, y) = d(y, x)$.
- 4° $d(x, z) \leq d(x, y) + d(y, z)$.

Property three is the *symmetry* property; property four is called the *triangle inequality*.

The second case is similar, in that the function is defined on the space crossed with itself, but the results are algebraic in nature instead of analytic.

Definition: Let S be a topological space, and define a function $\cdot : S \times S \rightarrow S$. If the function \cdot is both continuous and associative, then S is said to be a *topological semigroup*. Many of the standard fuzzy set theoretic and probabilistic operations can be expressed as operations in a topological semigroup, as can certain classes of operators in physics.

In the third case, the major problem with a continuous function as a preserver of topological structure is that the preservation is only one way, from the codomain to the domain. In order to be able to say that two spaces are topologically equivalent, the topological properties must be preserved in both directions, and, in addition, there must be some assurance that the spaces look somewhat alike as sets (since the definition includes both the set and the topology). These requirements force the definition of *homeomorphism*, defined as follows.

Definition: Let (S, τ) and (T, σ) be topological spaces. A function $f : S \rightarrow T$ which is a bijection, is continuous, and has continuous inverse is a *homeomorphism*. The spaces S and T are said to be *homeomorphic*.

The fourth case is that of a *projection*, a mapping from a product space to a subset of the coordinates, under the subspace topology. The fifth case is also called a projection, but occurs when a topological space T has an equivalence relation ρ defined on it. The projection is then the map $f: T \rightarrow T/\rho$ (where T/ρ is the space of equivalence classes of T under ρ) obtained by sending each element to its equivalence class.

Separation Axioms

As noted above, a homeomorphism preserves all topological properties between spaces. There are many types of properties that are topological in nature, such as connectedness, completeness, and compactness. One topological idea of importance in the main portion of this work is that of *separation*, or a description of exactly how points in the space can be distinguished. There are five basic classes of separation, denoted T_0 through T_4 . Each class T_i contains T_{i+1} . The largest class, T_0 , is not all inclusive, so there are actually six classes, the non- T_0 , and the T_0 through T_4 . Each of these classes will be defined. In each definition, (S, τ) is a topological space, and $x, y \in S, x \neq y$.

Definition: S is said to be T_0 if at least one of x, y has a neighborhood which does not contain the other. S is T_1 if each of the points x and y has a neighborhood which does not contain the other point.

These two properties are extremely weak properties. Most topologists do not work with spaces which have only T_1 or less separation. For general use, most topologists require that the space be at least T_2 , or *Hausdorff*.

Definition: A space is T_2 if each of x and y have non-intersecting neighborhoods, that is, there is U , a neighborhood of x , and V , a neighborhood of y , such that $U \cap V = \emptyset$.

Definition: A T_2 space is T_3 , or *regular*, if each point x and closed set C have non-intersecting neighborhoods. The space is T_4 , or *normal*, if the point x can be replaced by an arbitrary closed set D , and still have non-intersecting neighborhoods exist.

A major result of topology states that:

Theorem: Every metric space is normal.

Since it was shown earlier in this work that the p -norm space of Salton, Fox, and Wu is non- T_0 , it must be the case that no metric can be defined on it. This would seem to indicate that the definition of similarity as used in this model is inadequate, from a topological point of view.

Vita

Steven C. Cater is a native of Arkansas, although most of his life has been spent in Louisiana. As an undergraduate at Louisiana State University, he was a 1971 Alumni Scholar, and was a recipient of the Pi Mu Epsilon Senior Mathematics Award in 1978. He received a BS in mathematics from Louisiana State University in 1978, and followed this with an MS, also in mathematics, from Louisiana State University in 1981. Since 1981, he has been an instructor in the Department of Computer Science, while working toward the PhD in computer science.

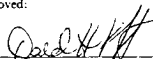
DOCTORAL EXAMINATION AND DISSERTATION REPORT

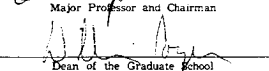
Candidate: Steven C. Cater

Major Field: Computer Science

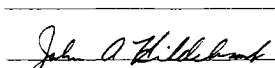
Title of Dissertation: The Topological Information Retrieval System and the Topological Paradigm: A Unification of the Major Models of Information Retrieval

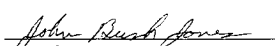
Approved:

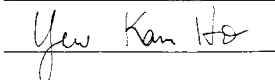

Major Professor and Chairman


Dean of the Graduate School

EXAMINING COMMITTEE:


Edward T. Lee


Duncan A. Buell



Date of Examination:

November 7, 1986