This is to certify that the

dissertation entitled

ONTOLOGY-BASED SIMILARITY
FOR CLUSTERING IN TEXT SPACE

presented by

Nasser Assem

has been accepted towards fulfillment
of the requirements for

Doctoral degree in Computer Science &
Engineering

B Ramamurthy
Major professor

Date 05-10-02

Ontology-based Similarity for Clustering in Text Space

By

Nasser Assem

A Dissertation

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

Doctor of Philosophy

Department of Computer Science & Engineering

2002

ABSTRACT

ONTOLOGY-BASED SIMILARITY FOR CLUSTERING IN TEXT SPACE

By

Nasser Assem

With the advent of the World Wide Web, and the increasing popularity of web search engines, there has been a renewed interest in information retrieval systems. In this research, we introduce a system that combines category-based and keyword-based concepts for a better information retrieval system. For improved document clustering, we proposed a document similarity measure that is based on keyword frequency in documents, but also uses an input ontology. This ontology is domain specific and includes a list of keywords organized with their degree of importance to the categories of the ontology. We evaluated the performance of this similarity measure and compared it to the standard cosine vector similarity measure. For that, we used document data with pre-determined structure as well as actual web documents. We designed a framework to generate synthetic data to model documents, and analyzed statistical attributes of documents in high dimension. For synthetic data analysis, we designed a controllable structure using various distributions of angle to specify cluster compactness and angle based inter-cluster overlap to specify cluster isolation. We address

the issue of modeling text documents, and propose the use of a graph data model that is based on the concept of semantic groups. We present a mechanism by which semantic groups can be used with document processing.

To my family

ACKNOWLEDGMENTS

I would like to thank Dr. Sakti Pramanik for being my guidance committee chair. Thanks also go to the other members of my guidance committee: Dr. Abdol H. Esfahanian, Dr. Eric Torng, Dr. Shawnee Vickery and Dr. George C. Stockman. Special thanks go to Dr. Wayne Dyksen for his support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

1.1 Motivation

With the advent of the World Wide Web, there has been a renewed interest in Information Retrieval systems. Most current WWW search engines are based mainly on keyword-based search. Some words may be conceptually different, even though they are syntactically similar and vice versa. Therefore, keyword-based search systems may retrieve documents which are not relevant to a query, or may not retrieve all relevant documents. To address this issue, some search engines also include category based search. However the two search mechanisms are usually used separately.

The space of category-based search systems can be characterized as a continuum. On one end the categories are automatically generated; such systems are fast and capture all possible categories. However, some categories may not necessarily make sense to users; the number of categories may also be huge. On the other end of the continuum, the categories are created by humans. Such categories fit perfectly

in the context of user searches, since they are most likely to be understandable by users. However, the process of creating and maintaining categories in such systems is slow and costly. The categories may not always be up-to-date. Moreover, the categorization system may not be global, and may have a low coverage rate, that is, only a small subset of documents can be covered by such categorization. For example, in this continuum, the Yahoo [18] categorization system is located at the extreme end of "manual" systems, since the categorization is made manually off-line. More recent search engines, like Lycos [50], have been augmented with semi-automated category systems such as WiseWire. However, they still require human intervention.

A search engine will be more user friendly and easy to maintain if there is a system that automatically generates and uses human-like categories. As shown in Figure 1.1, conceptually such a system would consist of three layers: (a) a top layer that contains human understandable categories, (b) a middle layer that contains automatically generated categories, and (c) a bottom layer that contains references to actual web pages. The top layer can be constructed based on a general categorization method. One of the main advantages of such system is that different levels of categorization offer the user the best of both worlds. On one hand, the top layer provides the user with concise closely related and human readable query results which are as easy to browse as Yahoo's manual categories. On the other hand, the middle and bottom layers offer the opportunity to access broader results. Another advantage is that automatic categorization minimizes human intervention; and hence alleviates system maintenance and update.

This work contributes to the middle layer by using ontology-based similarity mea-

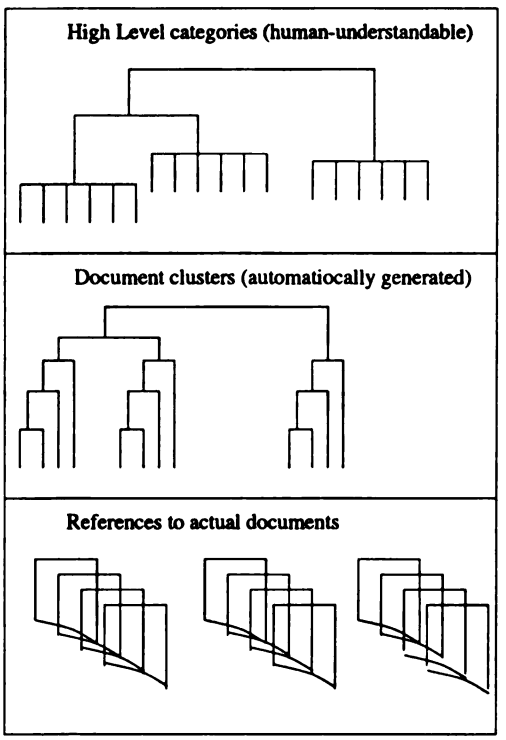


Figure 1.1: Architecture layers

sure that combines category and keyword based information, to generate document clusters with improved quality. It also proposes a graph data model for representation of documents.

1.2 Research Work

We have designed a technique that integrates text categorization with the standard keyword based search and produces improved document clustering. To design such technique, we have investigated several distance measures for the domain of text documents. Text categories used in this work are composed of a set of keywords with varying weights, where the weights determine how important the keyword is to

a category. The motivation for usage of an ontology-based weight is that keywords that are part of the same category should mutually contribute with a higher effect on the similarity of documents in which they appear. Combining the ontology with standard keyword-based similarity, adds a new semantic to the information retrieval system.

Clustering is used to perform document categorization. We present a clustering technique based on the ontology-based similarity measure, thus improving the quality of clusters. As previous work suggests that the quality of hierarchical clustering methods depends on the application domain, we analyze the effect of the types of data distributions on the performance of the similarity measure.

To achieve control of the parameters of synthetic data, such as angle distance between data centroids, we investigate several methods to model and generate synthetic documents.

We have proposed the use of a graph data model [64] for modeling documents. This model provides a general framework for mapping clusters to high level categories, thus allowing for better presentation of documents. This model also allows for better update of the underlying structure of documents.

1.3 Dissertation Outline

This dissertation is organized as follows. Chapter 2 introduces background information and related research work in the area of document processing, modeling, and clustering techniques, as well as on graph data models. In Chapter 3 we define the

proposed ontology-based similarity measure, and elaborate on how to evaluate its performance. Details on generation and characterization of synthetic data used for data distribution analysis are presented in Chapter 4. Chapter 5 focuses on the semantic groups of the graph data model suggested to use with documents. Chapter 6 summarizes the work of this dissertation and concludes with suggestions regarding future work.

Chapter 2

Background and Literature Review

2.1 Introduction

This chapter presents background information and related work on document modeling and clustering techniques and the graph data model. The concepts of term weighting and similarity measure are introduced. Different data and document clustering techniques are described. And since the graph data model provides a framework for modeling document clusters and categories, an introduction to graph data models is provided.

2.2 Document Modeling and Processing

Information Retrieval systems have been developed to help manage the huge volume of literature documents. Documents are generally represented by a set of index terms or keywords. A logical view of a document is its representation by the set of all the words

it contains; however, this may require a lot of storage space for the whole collection of documents. That is why, in general, documents are first pre-processed after they are parsed to identify all terms. Some words like articles and connectives (also called stop-words) are eliminated. Some words are reduced to their common grammatical root; this process is called stemming. Figure 2.1 shows a typical preprocessing sequence of a document from full text to index terms for document modeling.

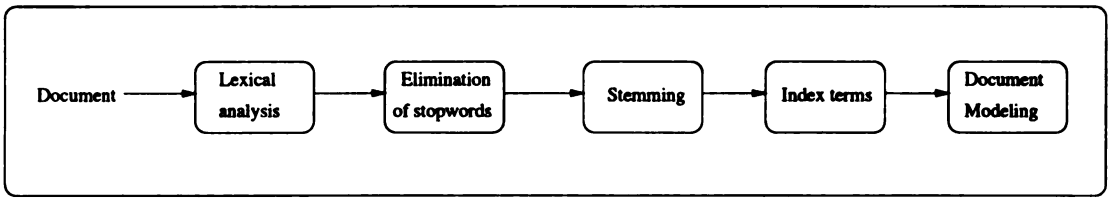


Figure 2.1: Document preprocessing: from full text to index terms

2.2.1 Document Preprocessing

Document preprocessing is the preparation of the documents for modeling, retrieval and access. The processing consists of the following operations: (1) lexical analysis, which identify words; (2) elimination of stop words, which filters out words with low discrimination value; (3) stemming, which is removing affixes (suffixes), allows retrieval of documents containing static variations; (4) construction of term categorization structures, or "Thesaurus", which can be used for the construction of ontologies.

This processing is made to improve the quality and speed of document and data retrieval. Not all operations are performed for every data retrieval system; it mainly depends on the domain of use and the cost of using it. The following subsections

describe each of the above operations.

Lexical Analysis This is the process of extracting a stream of selected words from a stream of characters. To do this, it is required to identify the words of a document. Punctuation marks and single letters are usually removed.

Elimination of Stop words Words that occur in most documents like "and", "or" should be removed. They are considered useless because they do not have a high discrimination value. Their elimination also reduces the size of the document database. A list of stop words can be found in [22].

Stemming Sometimes two different but related forms of a word exist in documents like plurals, and various verb forms. Stemming is the process of reducing a word to its origin. A stem is the portion of the word remaining after removing its affixes. This can be achieved using a finite state machine that is based on linguistic rules. This operation also reduces the size of the document database.

Thesauri Construction is the method of creating a standard vocabulary, which provides classified hierarchies. A thesaurus consists of a compiled list of important words in a given domain of knowledge. For each word in the list, a set of related words is defined. A term in the thesaurus is usually a single word. Using a Thesaurus in the data retrieval allows for adding concept to words, which significantly increases the quality of clustering. This doesn't always work well, because relationships captured in the thesauri frequently are not valid in the local context of a data set. For example,

general relationships between keywords may not be very helpful in a domain specific data set, like a medical database, as opposed to a more technical set of relationships, like those between brand names and generic names.

2.2.2 Document Modeling

This defines a framework in which documents are represented by logical views based on the information that the preprocessor extracts from the documents. It also defines how a similarity measure between documents is quantified. The main models that are proposed in the literature are the vector model [69], the boolean model and the probabilistic model [68, 70]. The boolean model uses a binary weight to represent documents. The relationship between the document and keywords is basically whether the keyword is cited in the document or not. The vector model uses a more involved weight system than binary weights to index terms in documents, allowing for partial matching by computing the degree of similarity between documents. This allows for a possible ranking of retrieved documents in decreasing order of degree of similarity. This is especially useful for a system where range queries are used. The probabilistic model iteratively defines the relationship between documents and keywords using some initial formula estimates that are improved in a multi-step fashion based, for instance, on input from the user. In the following we focus on the vector model.

Vector Representation of Documents: The weight w_{ij} associated with a term k_j and document d_i is positive and non binary, and the vector for a document d_i is:

$\vec{d_i} = (w_{i1}, w_{i2}, \ldots, w_{it})$, where t is the number of index terms in the system. The number t of keywords can be computed based on the preprocessor output by removing the keywords from a predefined stop-list, or by including only those terms that do not appear in too many documents; a threshold can be defined for this purpose. Another issue that is addressed here, as well as while preprocessing documents, is what form of the keywords should be used; stemming may not necessarily improve the quality of the representation of documents.

Term-Weighting System: Keywords that are frequently mentioned in a document are useful to categorize this document. Therefore, a term frequency, which is the number of times a keyword occurs in a document, should be used as part of the term-weighting system.

As proposed in [69], the widely adopted term-weighting scheme is as follows. If $freq_{ij}$ is the raw frequency of term k_j in the document d_i, that is the number of times the term k_j is mentioned in the text of the document d_i, then, the normalized frequency f_{ij} of term k_j in document d_i is: $f_{ij} = \frac{freq_{ij}}{\max_k freq_{ik}}$, where the maximum is computed over all terms of document d_i.

Standard term weighting schemes also take into account the inverse document frequency, also called the *idf* factor: $idf_j = log\frac{N}{n_j}$, where N is the total number of documents in the system, and n_j is the number of documents in which the term k_j appears. Terms that appear in many documents are not very useful for distinguishing documents. In these term-weighting schemes, the weight associated with d_i and k_j is: $w_{ij} = f_{ij} \times idf_j$.

The normalized frequency allows for a well defined domain, where a raw weight can be a real number between 0 and 1. The second term in the above formula, also called the *idf* term, allows for terms that are not being used in too many documents to have more contribution in distinguishing those documents in which those terms appear; this is specially useful for domain specific terms, that are in general not widely used outside their domain. A user would definitely try to use those keywords to retrieve the documents that he/she wants.

Similarity Measure Between Documents A similarity measure between two documents is based on the number of keywords that belong to both documents [69].

The degree of similarity of documents d_i and d_j is the correlation between the vectors $\vec{d_i}$ and $\vec{d_j}$, which can be quantified by the cosine of the angle between the two vectors.

Let's consider that there are t different keywords in the collection of documents. If a document d_i is represented by the following vector in the t-dimensional space: $\vec{d_i} = (w_{i1}, w_{i2}, w_{i3}, \ldots w_{it})$, then the following length can be used:

$$|\vec{d_i}| = \sqrt{\sum_{k=1}^{t} (w_{ik})^2} \tag{2.1}$$

and the similarity between two documents d_i and d_j can be defined by the *cosine vector* similarity measure:

$$sim(d_i, d_j) = \frac{\sum_{k=1}^{t} w_{ik} \times w_{jk}}{\sqrt{\sum_{k=1}^{t} (w_{ik})^2} \times \sqrt{\sum_{k=1}^{t} (w_{jk})^2}} \tag{2.2}$$

The numerator of the above formula obviously is useful in associating two documents if they share many common keywords. Those keywords not shared by both documents are eliminated since the product would be zero. The denominator of the formula, which is basically a normalizing factor is used to avoid discriminating against small documents in favor of large documents. A normalization factor should be incorporated into the similarity measure to normalize the length of the document vectors. This way, relevant short documents have equal chance to be clustered as larger documents.

Comparison of Vector Model to Other Models The probabilistic model and the boolean Model are the main alternative models used in Information Retrieval Systems. The vector model is superior to the boolean model, and at least almost as good as the probabilistic model.

The advantages of the vector model can be summarized as follows. First, the simple term-weighting scheme improves retrieval performance, since the similarity measure can be easily implemented, unlike the probabilistic model, where an iterative process is needed. Second, the partial matching strategy allows for the retrieval of approximately similar documents. This is useful for range queries, where exact match is not realistic, or is unnecessary. Finally, the cosine ranking formula sorts the documents according to their degree of similarity; this is useful for ranking the documents retrieved for a given cluster.

One of the disadvantages of the vector model is that the index terms are assumed to be mutually independent; which is not necessarily true, since some terms usually

appear together in a lot of documents, and may therefore be correlated. The generalized vector space model addresses this issue. However, it has not been shown that the increase in complexity does improve performance compared to the standard vector model. Another disadvantage is the sparse vectors used for representing documents, with too many zeros; a lot of space may be wasted. However, some search data structures and algorithms are designed to take into consideration this issue.

2.3 Data and Document Clustering

A clustering is a type of classification of a set of objects [34]. There are hierarchical and partitional clustering methods. A hierarchical classification is a nested sequence of partitions. A partitional classification is a single partition. Hierarchical clustering is usually used to cluster documents, because it provides a hierarchy of clusters, that can be navigated further.

2.3.1 Hierarchical Clustering Techniques

A hierarchical clustering method is a procedure for transforming a similarity matrix into a sequence of nested partitions. The resulting clustering is a sequence of partitions in which each partition is nested into the next partition in the sequence. An agglomerative algorithm for hierarchical clustering starts with the disjoint clustering, which places each of the documents in an individual cluster. The similarity matrix is used to merge two of these clusters, thus nesting this clustering into a second partition. The process is repeated to form a sequence of nested clusterings in which the number

of clusters decreases as the sequence progresses until a single cluster containing all documents remains. A dendrogram of a hierarchy of clusters can be obtained. It can be used to visualize and assess to some degree the resulting clusters [74]. The main hierarchical clustering methods in use are: (i) complete link: chooses that pair of clusters with maximum distance, (ii) single link: chooses the minimum distance, and (iii) average link: chooses the average distance. The complete link method has been widely used for document clustering; we describe an algorithm for it in the following section.

2.3.2 Complete Link Clustering Algorithm

A matrix updating algorithm, suggested by King [39] and made popular by Johnson [37], is used to implement complete link clustering method. The algorithm, as described in [34], is as follows.

Suppose that we have n documents, and let the nxn matrix $D = [d(i,j)]$ be the similarity matrix. The clusterings are assigned sequence numbers $0, 1, ..., (n-1)$ and $L(k)$ is the level of the kth clustering. We denote the similarity between clusters (r) and (s) by $d[(r),(s)]$.

Step 1. Begin with the disjoint clustering having level $L(0) = 0$ and sequence number $m = 0$.

Step 2. Find the least dissimilar pair of clusters in the current clustering, say pair $\{(r),(s)\}$, according to $d[(r),(s)] = min\{d[(i),(j)]\}$ where the minimum is over all pairs of clusters in the current clustering.

Step 3. Increment the sequence number: $m \leftarrow m + 1$. Merge clusters (r) and (s)

into a single cluster to form the next clustering m. Set the level of this clustering to $L(m) = d[(r), (s)]$

Step 4. Update the proximity matrix, D, by deleting the rows and columns corresponding to clusters (r) and (s) and adding a row and column corresponding to the newly formed cluster. The proximity between the new cluster, denoted (r, s) and old cluster (k) is defined as follows[1]: $d[(k), (r, s)] = max\{d[(k), (r)], d[(k), (s)]\}$

Step 5. If all objects are in one cluster, stop.

Else, go to step 2.

2.4 Graph Data Model and Documents

Text documents, in general, and web documents, in particular, do not have a well defined structure or schema. In fact, in [1, 12], web documents are referred to as semistructured data. Therefore, it is not easy to model them using a classical data model, like the relational data model. To model documents and their clusters and categories, we propose to use a graph data model [64] with a novel idea of semantic group. The graph data model used has a structure similar to the ER (Entity-Relationship) data model [14], the Hypernode data model [44, 62], and the graph data model proposed in [41]. The basic structure of the data model used in this research is a graph where nodes correspond to entities, and links correspond to relationships between these entities. The links are labeled, and their labels define types of relations.

[1]For single link method, the proximity is updated in step 4 above as follows:
$d[(k), (r, s)] = min\{d[(k), (r)], d[(k), (s)]\}$
For average link method, it is updated as follows:
$d[(k), (r, s)] = avg\{d[(k), (r)], d[(k), (s)]\}$

In traditional database models, such as the relational data model, views are built on top of base relations. In the graph data model, the definition of views is conceptually different. Views are built as part of the data itself. Views are subgraphs in which relationships are logically chained. Two fundamental issues addressed in the graph data model are data sharing and view update. Links shared by multiple views cause the problem of updating these views independently; that is, an update on a given view may affect other views in an unwanted way.

In the following we will define the major components of a graph data model.

2.4.1 Graph-Based Database

A graph data model is a set of typed nodes (atomic information) connected with typed links (relations between nodes). A domain (subgraph), can also be abstracted by a node.

A graph-based database state can be defined by the 7-tuple:

$$DB =< N, NT, NM, L, LT, LM, D >,$$

where N is the set of all nodes; it represents a collection of atomic pieces of information. NT is the set of node types; every node has a type. NM is the mapping between nodes and their types. L is the set of links; it represents the relationships between atomic entities (nodes). LT is the set of link types; every link has a type. LM is the mapping between links and their types. D is the set of domains.

2.4.2 Domain

The last element in the tuple defining the state of a graph-based database is D, set of domains. A domain d is basically a subgraph; $d = \{c_i\}$, where the c_i are components (nodes and links) that belong to domain d. The notion of domain is close to the notion of traditional views, i.e a part of a database selected according to some criteria. However, data in domains are independent, and modifications in one domain are not reflected in others, even if they contain the same components.

A domain can be encapsulated into a node, which defines a composite structure similar to the Hypernode model [62]. Encapsulation of domains is recursive, i.e a domain may contain other domains.

2.4.3 Semantic Group

The semantic group concept captures transitivity of relationships in a database. Traditional views are also used to capture transitivity. Views can be created for different contexts in the database. However, views are defined on top of actual data, and therefore, update of views in traditional databases is a complex problem. It will be shown in the proposed data model, that transitivity of relations is naturally supported by the notion of semantic groups. Only relationships in the same semantic group are transitive.

2.5 Literature Review

This section presents an overview of literature work on document modeling and clustering techniques, and graph data models. Section 2.5.1 presents some work done in document modeling. Section 2.5.2 contains a survey of data and document clustering techniques. Section 2.5.3 presents a review of work done in graph data models.

2.5.1 Document Modeling and Similarity Measure

In this section we present related work done in the area of document preprocessing, modeling, term-weighting and similarity measure between documents.

Salton [69], Frakes and Baeza-Yates [22] are among the classic references in the area of Information Retrieval. To extract the most useful keywords from documents, the preprocessing phase of Information Retrieval Systems usually includes the removal of useless words, also called stop-list, from documents, as proposed by Fox [20]. The preprocessing phase of some Information Retrieval Systems also includes normalization of keywords, called document stemming, as proposed by the Porter algorithm [61, 21].

In most recent Information Retrieval systems, documents are modeled using the vector space as proposed by Salton [69], where each document is represented by a vector of keywords. The product of term frequency (tf) and the inverse document frequency (idf) has been proposed as term-weighting scheme.

Different similarity measures between documents and keywords can be used to determine how close documents are to each other. The cosine vector similarity measure

is widely used [8, 22, 69, 77].

2.5.2 Data and Document Clustering

In this section we present related work done in the area of data and document clustering. Murtagh [57] presented a survey of hierarchical clustering algorithms. The study focused on performance analysis of those algorithms with respect to time and space complexity; not much comparison of cluster validity was addressed. Another survey of hierarchical clustering methods is presented in [34]. It was suggested that no hierarchical clustering method is best for all clustering problems, and that the quality of hierarchical clustering methods depends on the application domain. It was also noted that with single link clustering method, clusters easily chain together and are less compact, since only a single small edge between two large clusters is needed to merge the clusters. The complete link clustering method is more conservative; in that, all pairs of patterns must be related before the patterns are clustered, resulting in more compact clusters.

Recent work using hierarchical clustering has been done in the area of cellular manufacturing for identifying alternative cell designs. Measures of association for forming part of families and machine cells were defined, and different hierarchical clustering methods were compared. Mosier [56] showed that complete link dominated single link for cluster recovery; and that single link was best overall for between-cells move measure. The study also showed that there are significant interaction effects between clustering techniques, similarity measures and data structures.

In [71], a comparison was performed between single, average and complete linkage

clusterings for a new similarity measure for the cell formation problem. The study showed that clustering techniques have greater impact than similarity measures, and there is no significant interaction between similarity measures and clustering techniques. Complete linkage clustering method performed best, and average linkage clustering performed worst. The authors introduced a new similarity measure that has the desirable property of yielding a perfect 1.0 whenever one part requires the same set or subset of the machines required by another part.

Vakharia and Wemmerlov [75] showed that while the choice of clustering techniques is more critical than the choice of similarity measure, differences among clustering techniques (due to chaining tendencies) can be reduced by restricting the solution space for acceptable cell configurations. Complete link performed poorly. Average link performed best.

Some implementations of hierarchical clustering algorithms have been proposed in [19]. In [17] a sequential, agglomerative, hierarchical, non overlapping (SAHN) clustering algorithm was described.

Hierarchies have been used in Information Retrieval systems such as in [9, 23, 43], for organization, summarization and finding topics. Clustering has been applied to documents in [10, 51, 67, 76, 81, 80].

An evaluation of topic-driven web crawlers has been done in [52]. Iwazume et al. in [33, 32] proposed the use of an ontology for document navigation and text categorization. This is different from our approach, where we do not use ontology separately, but combined with keyword-based similarity measure.

WordNet [54, 55] is a lexical database, where words are organized by different

types of relationships. WordNet thesauri can be used as an input ontology to our clustering technique. In [77], related work has been done, where words are clustered to produce such relationships. Other Artificial Intelligence approaches to classification of documents have been reported in [66, 6, 35, 47, 45, 46, 48, 78] where machine learning is emphasized.

SQLEM [58] is an SQL implementation of a clustering algorithm, claimed by the authors to handle high dimensional data. "Scatter/Gather" [29, 30, 60] is a novel idea proposed to cluster query results documents, before presenting them to the users. In [26], [5] and more recently in [3, 4], efficient clustering algorithms have been proposed for large databases.

2.5.3 Graph Data Model and Semistructured Data

In this section we discuss some graph-based data models, and we also present some relevant work that deals with modeling data and the Web.

The graph data model that is used in this research has a structure similar to the ER (Entity-Relationship) data model proposed by Chen [14], the Hypernode data model [44, 62], and the graph data model proposed by Kunii in [41]. In the ER model, nodes represent entities and links represent relationships between them. The basic structure in the hypernode model is defined by $G = (N, E)$, where G is the (unique) label of the hypernode, and (N, E) is a directed graph, whose nodes can be labels of other hypernodes (directed graph), attribute names, or atomic values. Since a node can itself be a nested graph, this model can support complex objects.

The GRAS data model presented in [38] relies on attributed graphs. In this

model, objects are represented by typed nodes which may carry attributes. Relations between objects are modeled by bidirectional edges (with no attributes). The authors used graph schemes to represent the semantics of a database. This data model was biased toward modeling software engineering systems. Unlike in GRAS, in our data model, both the schema and instance of the database are simultaneously represented. This results in a more dynamic update of the database.

GOOD [27, 24, 59] started as a database interface, then evolved as a graph object oriented database system. Actually, it is a graph representation of an object oriented database. Nodes represent objects, and links represent relationships between objects. Here again, the authors distinguished between an object base instance, and an object base scheme.

The Hy^+ system [15, 16] is a query and visualization system that provides the user with tools that assist in forming queries, and visualizing results. It is not a full graph data model. In [49] the authors presented a graph-based object model and focused on visual retrieval and user interface functionalities. Another graph-based framework for visual access has been proposed in [13].

Griffin et al. [25] used incremental view update through auxiliary tables that contain information recorded since the last view update. This work was motivated by performance improvement. It is different from our approach, where we address view update in the context of semantic groups.

Related work has also been conducted in the area of semi-structured data [1, 2, 11, 12]. Theoretical issues related to modeling and querying such data were addressed. The Object Exchange Model (OEM) was compared in [1] to relational and object

data models. The OEM model was shown to be more appropriate for modeling semi-structured data. In [12] the query language UnQL was proposed for querying data organized as a rooted graph. Another system based on the OEM model called TSIMMIS [28] was proposed to extract semi-structured data from HTML documents using template matching.

Other work has been done for integrating and querying data over the Web. A WWW resource database system [79], that uses limited hypertext structure in indexing HTML documents was developed. Some SQL-like languages, such as WebSQL [53] and W3QS [40], considered both hypertext structure querying and content querying. In our approach, we consider modeling semi-structured data, thus we use a graph based query language rather than a SQL-like language.

Other web querying systems such as [7] and [42] are based on a logic data model. The ADM model [7] was developed to manage and restructure Web data using an interesting page-oriented approach that considers HTML forms. The authors provided transformation mechanisms from hypertext to relational views.

2.6 Summary

In this chapter we introduced the concepts of document modeling, clustering techniques and graph data models. These concepts will be used to describe the research work performed for this dissertation. We have also described the published research work related to document modeling and clustering techniques, and graph data models.

The reviewed papers suggest that no specific clustering method is best for all clus-

tering problems and that the quality of clustering methods depends on the application domain.

We observe that similarity measure based only on keyword frequency is not appropriate for all domains. In particular, if there is an ontology available for a specific domain, the use of that ontology can provide some added semantics that can improve document clustering.

It is worth mentioning that most studies on graph data model focused on defining new approaches to presenting a graph data model. Our work, on the other hand, focuses on using views through the notion of semantic groups for modeling and querying documents, and addresses the issue of updating views in graph data model.

Chapter 3

Ontology-based Similarity Measure

3.1 Introduction

As mentioned in Chapter 1, similarity measure based only on keyword frequency is not appropriate for all types of data sets. In particular, if there is an ontology available for a specific domain, keyword-based similarity measure may not take advantage of the semantics that can be provided by this ontology.

Section 3.2 contains an analysis of cosine vector similarity and Euclidean distance measures for the domain of text documents. Section 3.3, focuses on the ontology-based similarity measure proposed in this work. It also describes the ontology in text space, and provides a general description of two approaches used to evaluate the performance of the proposed ontology-based similarity measure. The first approach uses a set of documents having pre-defined clusters, with a priori knowledge about its cluster structure. The second approach uses real document data sets, gathered from the web, without considering a priori knowledge about the structure of the data.

For the purpose of comparison, we provide for both approaches, results achieved using ontology-based similarity measure as well as using the standard cosine vector similarity. Those results are reported in Sections 3.4, and 3.5. Finally, in Section 3.6, we summarize this chapter.

3.2 Similarity and Distance Measures for Documents

While designing our proposed ontology-based similarity measure, we analyzed cosine vector similarity and Euclidean distance measures in Text Space. In this section, we discuss the relationship between standard cosine vector similarity measure and the Euclidean distance. These relationships lead us to focus on the cosine vector similarity measure and try to improve it.

3.2.1 Cosine Vector Similarity Measure

As seen in Chapter 2, in the case of a t-dimensional space, that is if the number of distinct keywords in the total collection of documents is t, and a document d_i in the collection is represented by the vector $\overrightarrow{d_i} = (w_{i1}, w_{i2}, \ldots, wit)$, then the cosine vector similarity between documents d_i and d_j is defined as:

$$sim(d_i, d_j) = \frac{\sum_{k=1}^{t}(w_{ik} \times w_{jk})}{\sqrt{\sum_{k=1}^{t} w_{ik}^2} \times \sqrt{\sum_{k=1}^{t} w_{jk}^2}} \tag{3.1}$$

Salton in [69], gave a motivation for the cosine vector similarity measure by arguing that the sum of the products of the frequency of corresponding keywords (or vector dot product) between two documents is a good indicator of the similarity between those documents. He then argued that, to avoid the problem of discriminating against smaller documents in favor of larger ones, a normalization by the length of the document vector is in order. Shapiro and Stockman in [72], invoking the Cauchy-Schwartz Inequality [73], also suggest the normalized dot product as a measure of similarity between vectors. We first set out to understand how will normalization affect document similarity measure.

Definition. Let d_i be any document represented by the vector $\overrightarrow{d_i} = (w_{i1}, w_{i2}, \ldots, wit)$, and let a be any positive real number. We define the scaling normalization of document d_i by a (scalar-document product) as the document $\frac{1}{a}d_i$, represented by the vector $\frac{1}{a}\overrightarrow{d_i}$. The kth component of the normalized vector is then:

$$w'_{ik} = \frac{w_{ik}}{a} \qquad k = 1, 2, \ldots, t \tag{3.2}$$

Proposition. The cosine vector similarity measure is invariant with respect to scaling.

Proof. Let d_i and d_j be any two documents represented by the vectors $\overrightarrow{d_i} = (w_{i1}, w_{i2}, \ldots, wit)$ and $\overrightarrow{d_j} = (w_{j1}, w_{j2}, \ldots, wjt)$ in the t-dimensional space, and let a and b be any two positive real numbers. We shall prove that

$$sim(\frac{d_i}{a}, \frac{d_j}{b}) = sim(d_i, d_j), \tag{3.3}$$

where $sim(d_i, d_j)$ is the cosine vector similarity between documents d_i and d_j.

$$
\begin{aligned}
sim(\frac{d_i}{a}, \frac{d_j}{b}) &= \frac{\sum_{k=1}^{t}(\frac{w_{ik}}{a} \times \frac{w_{jk}}{b})}{\sqrt{\sum_{k=1}^{t}(\frac{w_{ik}}{a})^2} \times \sqrt{\sum_{k=1}^{t}(\frac{w_{jk}}{b})^2}} \\
&= \frac{\frac{1}{ab}\sum_{k=1}^{t}(w_{ik} \times w_{jk})}{\sqrt{\frac{1}{a^2}\sum_{k=1}^{t} w_{ik}^2} \times \sqrt{\frac{1}{b^2}\sum_{k=1}^{t} w_{jk}^2}} \\
&= sim(d_i, d_j)
\end{aligned}
\tag{3.4}
$$

This result is another form of the property of normalized cross correlation being independent of scale factors [72].

3.2.2 Euclidean Distance Measure

Figure 3.1 illustrates the cosine vector similarity and the Euclidean distance in a 3-dimensional space, where the cosine vector similarity between documents d_i and d_j is represented by $\cos\alpha$, and the standard Euclidean distance by dis(d_i,d_j).

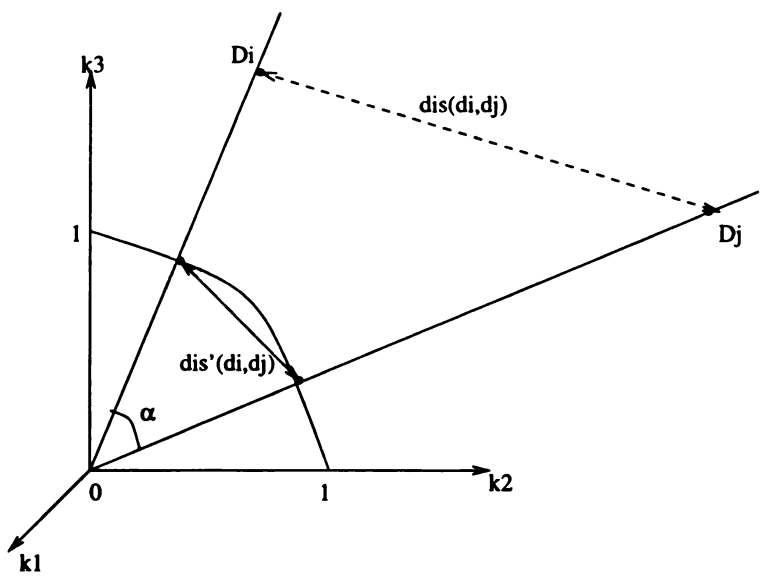


Figure 3.1: Illustration of distance and similarity measures

In the following, we investigate what would happen if the Euclidean distance is used to quantify the dissimilarity of documents. We are here talking about dissimilarity instead of similarity because as seen in Figure 3.1, the bigger the Euclidean distance between two documents on the surface of the unit hypersphere, the more they are dissimilar. After all, by using the Euclidean distance, both arguments presented by Salton in [69] remain valid. The sum of the squared difference between the frequency of corresponding keywords is an indicator of how dissimilar two documents are.

Standard Euclidean Distance

Let's consider that there are t different keywords in the collection of documents. If a document d_i is represented by the following vector in the t-dimensional space: $\overrightarrow{d_i} = (w_{i1}, w_{i2}, \ldots, w_{it})$, then the dissimilarity between two documents d_i and d_j can be defined by the Euclidean distance as follows:

$$dis(d_i, d_j) = \sqrt{\sum_{k=1}^{t}(w_{ik} - w_{jk})^2} \tag{3.5}$$

It has the following metric properties:

$$\forall i, dis(d_i, d_i) = 0 \tag{3.6}$$

$$\forall i, j | i \neq j, dis(d_i, d_j) > 0 \tag{3.7}$$

$$\forall i, j, dis(d_i, d_j) = dis(d_j, d_i) \tag{3.8}$$

$$\forall i, j, k, dis(d_i, d_j) \leq dis(d_i, d_k) + dis(d_k, d_j) \tag{3.9}$$

It has also the property of being invariant with respect to rotation and translation. Because of these properties, the Euclidean distance has been used as a measure of dissimilarity in many pattern recognition domains, like image retrieval. The problem with using the Euclidean distance for the domain of documents is that documents with comparable size tend to be more similar to each other and less similar to other much larger or much smaller documents, even though all these documents are semantically similar. In the following we tried to address this problem by normalizing the documents vectors by their vector length.

Normalization by Document Length

Let document d_i be represented by the t-dimensional vector $\vec{d_i} = (w_{i1}, w_{i2}, \ldots, w_{it})$. We define the length of document d_i as the length of vector $\vec{d_i}$, that is:

$$|d_i| = |\vec{d_i}| = \sqrt{\sum_{k=1}^{t} w_{ik}^2} \tag{3.10}$$

We define the normalized document vector as:

$$\vec{d_i}' = \frac{1}{|\vec{d_i}|}.\vec{d_i} \tag{3.11}$$

The kth component of the normalized vector is then:

$$w_{ik}' = \frac{w_{ik}}{|\vec{d_i}|} \qquad k = 1, 2, \ldots, t \tag{3.12}$$

The modified dissimilarity between two documents d_i and d_j can be defined by

the Euclidean distance between the document vectors normalized by their length as follows:

$$dis'(d_i, d_j) = \sqrt{\sum_{k=1}^{t}(w'_{ik} - w'_{jk})^2} \tag{3.13}$$

The normalized distance between d_i and d_j in Figure 3.1 is illustrated by dis'(d_i,d_j).

Proposition. The Euclidean distance applied to the normalized document vectors is semantically equivalent to the standard cosine vector similarity measure. In fact, the cosine vector similarity is quadratically proportional to the Euclidean distance.

Proof. Let d_i and d_j be any two documents represented by the vectors $\vec{d_i} = (w_{i1}, w_{i2}, \ldots, wit)$ and $\vec{d_j} = (w_{j1}, w_{j2}, ..., wjk)$ in the t-dimensional space. We shall prove that

$$sim(d_i, d_j) = 1 - \frac{dis'^2(d_i, d_j)}{2}, \tag{3.14}$$

where $sim(d_i, d_j)$ is the cosine vector similarity between documents d_i and d_j, and $dis'(d_i, d_j)$ is the Euclidean distance between the normalized document vectors of d_i and d_j.

$$
\begin{aligned}
dis'^2(d_i, d_j) &= \sum_{k=1}^{t} (w'_{ik} - w'_{jk})^2 \\
&= \sum_{k=1}^{t} \left(\frac{w_{ik}}{\sqrt{\sum_{l=1}^{t} w_{il}^2}} - \frac{w_{jk}}{\sqrt{\sum_{l=1}^{t} w_{jl}^2}} \right)^2 \\
&= \sum_{k=1}^{t} \left(\frac{w_{ik}^2}{\sum_{l=1}^{t} w_{il}^2} - \frac{2 \times w_{ik} \times w_{jk}}{\sqrt{\sum_{l=1}^{t} w_{il}^2} \sqrt{\sum_{l=1}^{t} w_{jl}^2}} + \frac{w_{jk}^2}{\sum_{l=1}^{t} w_{jl}^2} \right) \\
&= 1 - 2 \frac{\sum_{k=1}^{t} (w_{ik} \times w_{jk})}{\sqrt{\sum_{k=1}^{t} w_{ik}^2} \times \sqrt{\sum_{k=1}^{t} w_{jk}^2}} + 1 \\
&= 2 - 2sim(d_i, d_j)
\end{aligned}
\tag{3.15}
$$

This equation is another form of the cosine law in [73].

3.2.3 Discussion

In this section, we have shown that the Euclidean distance applied to the normalized document vectors is semantically equivalent to the standard cosine vector similarity measure. While the Euclidean distance has the property of being invariant with respect to rotation and translation, the cosine vector similarity measure has the property of being invariant with respect to scaling. In the domain of documents, the latter property is more important since, for example, we expect a technical paper or a news report to have a high similarity measure to their abstract or shorter versions. Moreover, with the advance in the development of ontologies and domain specific topical hierarchies, as in most recent Web search engines, the ability to automatically use the semantics provided by such knowledge structures is desirable. For those reasons, we

proposed a hybrid similarity measure that combines both keyword information and ontology-based knowledge. In the next section, we describe the proposed similarity measure, and report on its evaluation.

3.3 Proposed Ontology-based Similarity Measure (OBSM)

In this section, we introduce the use of ontology in text space in Section 3.3.2. We define the proposed ontology-based similarity measure (OBSM) in Section 3.3.3. Section 3.3.4 provides a general description of the two approaches used to evaluate the performance of OBSM. The first approach uses a set of documents having pre-defined clusters, with a priori knowledge about its cluster structure. The second approach uses real document data sets, gathered from the web, without considering a priori knowledge about the structure of the data.

3.3.1 Motivation

As discussed in the previous section, the use of ontology-based knowledge is motivated by the essence and intrinsic properties of the cosine vector similarity measure itself. Keywords that are part of a same category or topic should mutually contribute with a higher effect on the similarity of two documents in which they appear.

3.3.2 Ontology Used in Text Space

An ontology in the text space can be defined as a hierarchy of categories. Each category is defined by a set of keywords, ordered by their importance with respect to their category. This order is achieved by associating a weight that measures the importance of a keyword in that category. Keywords may be part of many categories. This allows for an overlapping hierarchical category structure. Categories are based on term to term relationships and not on their appearance is documents. The ontology relationships are obtained by considering the terms as part of concepts. Combining the ontology with standard keyword-based similarity, adds a new semantic to the information retrieval system. An ontology can be constructed from a term categorization structure such as a thesaurus, which generally includes synonyms. However, relationships between keywords in an ontology may be general. Ideally, they should be specified by a domain specific expert.

3.3.3 Ontology-based Similarity Measure

An ontology is used to highlight those keywords that may be important to some specific categories and not others. In this ontology, a category is defined as a set of keywords sorted by order of importance for that category. A weight is used to capture this order of importance. A keyword that happens to be significant to a category will be having a bigger contribution to the similarity between documents that contain it.

If t is the number of distinct keywords in the whole collection of documents, s is the number of categories in the ontology, w_{ik} is the term weight associated with

keyword k and document d_i, and w'_{ck} is the weight of keyword k ($k = 1, 2, \ldots, t$) with respect to category c, then the ontology-based similarity between two documents d_i and d_j, represented by the t-dimensional vectors $\overrightarrow{d_i} = (w_{i1}, w_{i2}, \ldots, w_{it})$ and $\overrightarrow{d_j} = (w_{j1}, w_{j2}, \ldots, w_{jt})$, is:

$$sim(d_i, d_j) = \frac{\sum_{c=1}^{s}(\sum_{k=1}^{t} W_{ick} \times \sum_{k=1}^{t} W_{jck})}{\sqrt{\sum_{c=1}^{s}(\sum_{k=1}^{t} W_{ick})^2 \times \sum_{c=1}^{s}(\sum_{k=1}^{t} W_{jck})^2}}, \tag{3.16}$$

where $W_{ick} = w_{ik} \times w'_{ck}$.

The original weight associated with keyword k and document d_i is multiplied by the relative weight of keyword k with respect to category c. The sum of the products is performed over all categories. Different keywords can be identified if they are in the same category. This guarantees that if there exists a category to which many common keywords of documents d_i and d_j belong to, then the contribution of those keywords is expanded, causing the two documents to be more similar to each other than to other documents. Therefore, those documents will have a better chance to be classified in a cluster that semantically resembles the category of the input ontology. It is worth noting that we keep the normalization by the vector length, so that again we avoid discrimination against smaller documents in favor of larger ones.

OBSM can also be used for the similarity between documents and queries in the case of interactive Information Retrieval systems as well, like a search engine. In that case, the user may even contribute interactively in the choice of the categories that may be used to expand the contribution of the keywords that he/she enters in the query. In the next sections, we present results of our evaluation of the performance

of OBSM.

3.3.4 Performance Evaluation Approaches

We considered two different evaluation approaches. In the first approach, we generate clustered data (here clustered is in terms of keyword overlap). This is similar to the training set approach used in artificial intelligence based clustering. Then we compute the distance between these clusters in terms of similarity measure, using standard cosine vector similarity measure (CVSM) and OBSM, and we check if the clusters are preserved. In the second approach, we collect real documents and cluster them using both similarity measures.

3.4 Data Sets With a Priori Knowledge

In this approach, to evaluate the performance of our technique using OBSM, we generated a set of documents, with keywords drawn from a limited set of keywords. An ontology is built associating keywords into different categories, with weights representing an ordering within a category. A document is then associated with a category by having more keywords drawn from that category than any other. We vary the overlap between documents in terms of the percentage of common keywords, both within the same cluster (intra-cluster overlap) and from different clusters (inter-cluster overlap). If a cluster i has n_i t-dimensional data points $\{\overrightarrow{d_{ik}}\}$ $(k = 1, 2, ..., n_i)$, then the

intra-cluster and inter-cluster average similarities are defined respectively as follows:

$$d_{ii} = \frac{1}{n_i(n_i - 1)} \sum_{k=1}^{n_i} \sum_{l=1, l \neq k}^{n_i} sim(d_{ik}, d_{il}) \tag{3.17}$$

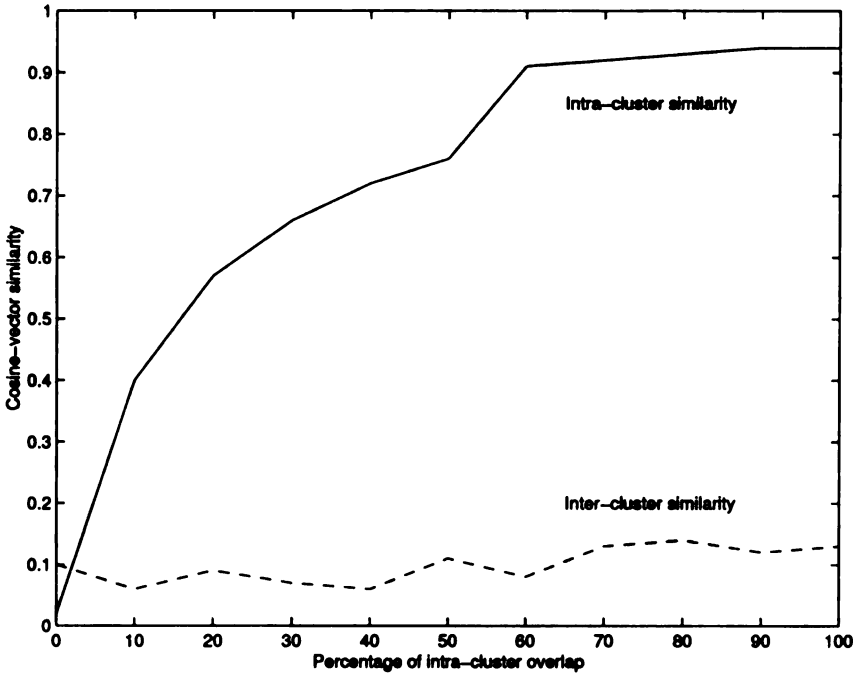
$$d_{ij} = \frac{1}{n_i n_j} \sum_{k=1}^{n_i} \sum_{l=1}^{n_j} sim(d_{ik}, d_{jl}) \tag{3.18}$$

For various data sets, we computed the intra-cluster and inter-cluster average document similarities. A data set with a good clustered structure should exhibit a high intra-cluster similarity, showing a good compactness of the clusters, and a low inter-cluster similarity, showing a good isolation between clusters.

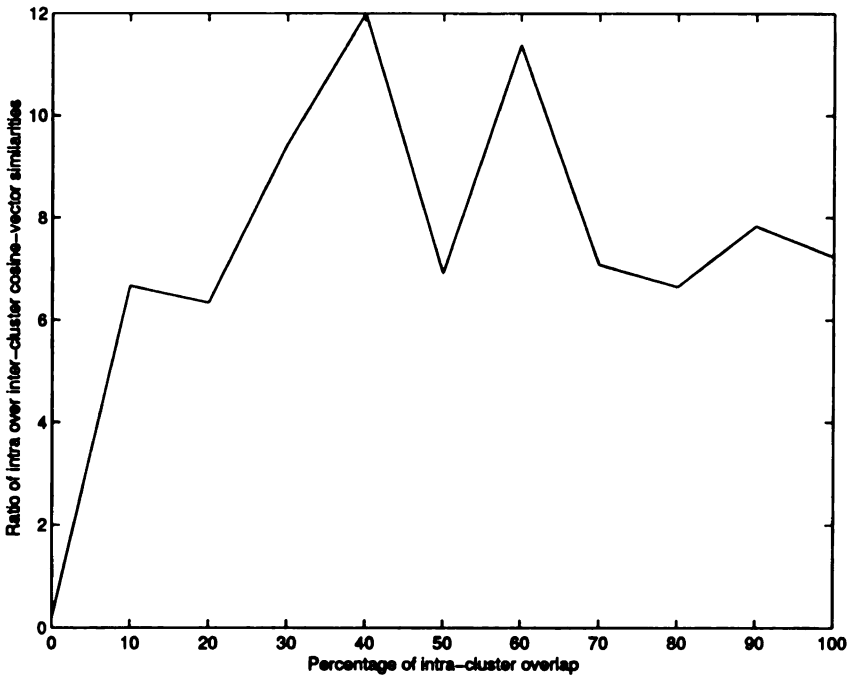
3.4.1 Varying Intra-cluster Overlap

Due to the correlation between categories used in the ontology on one hand, and the clusters used to generate the documents on the other hand, the ontology should be generated independent from the documents. First, we used a data set with a dimension of 20. The keywords have been drawn in a pool of two categories of 10 documents with an overlap of 10%.

The results show that OBSM reflects better the clustered structure of the documents. This is illustrated in Figure 3.2 and Figure 3.3. Figure 3.2(a) shows the variation of intra-category and inter-category CVSM similarities. Figure 3.2(b) shows the ratio of intra-category over inter-category CVSM similarity as a function of the intra-category overlap. We found that for this particular data set, the ratio of intra-cluster over inter-cluster CVSM similarity is not a non decreasing function of the



(a) Intra and inter-cluster similarities



(b) Intra over inter-cluster similarity ratio

Figure 3.2: Intra and inter-cluster CVSM similarities

Dimension (keywords)	1000
Number of categories	50
Number of documents	2000
Inter-category overlap	5%

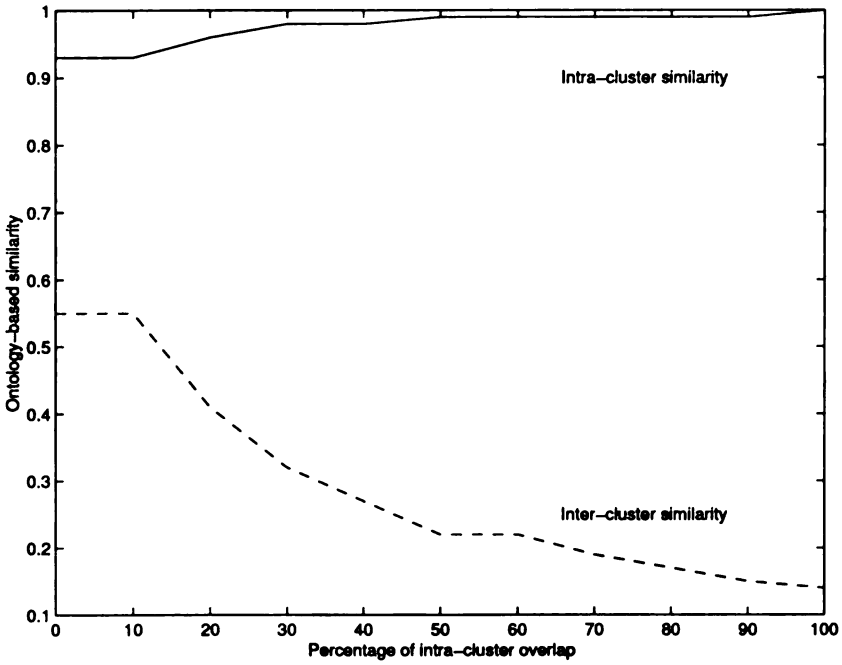
Table 3.1: Data set used to generate Figure 3.4

intra-category overlap.

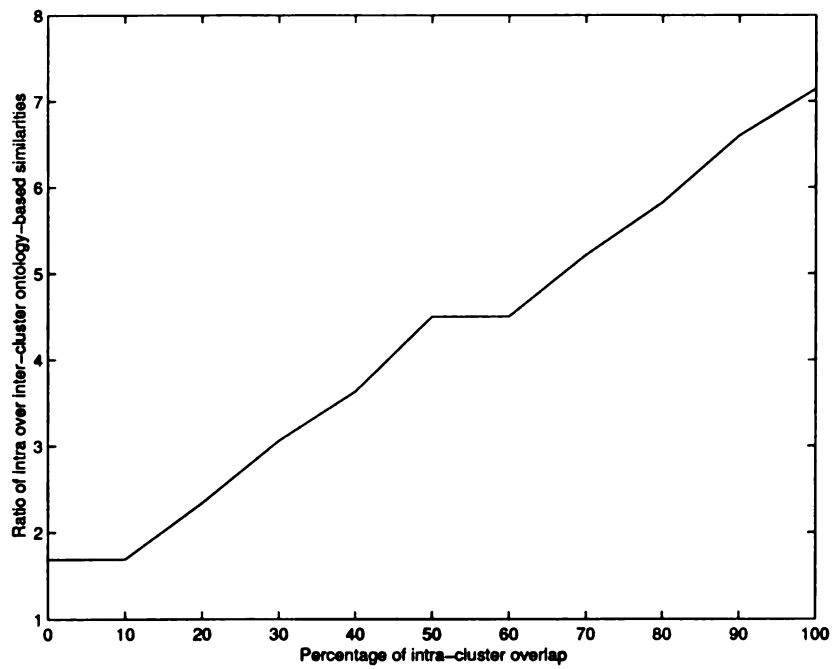
Figure 3.3(a) and Figure 3.3(b) show the same terms computed using OBSM. Unlike for CVSM similarity, the ratio of intra-cluster over inter-cluster OBSM similarity is a non decreasing function of the intra-category overlap. This means that for two data sets, one exhibiting more compact clusters than the other, OBSM will determine that the first data set has a better clustered structure. Whereas, using CVSM, that may not be guaranteed.

As a next step we have decided to generate ontology automatically. We generate documents varying: (i) the number of keywords; (ii) the number of clusters; (iii) the number of documents; (iv) the percentage of intra-cluster overlap; (v) the percentage of inter-cluster overlap. For different data sets, we compute the intra and inter-cluster average similarities, as well as their ratio, for both CVSM and OBSM.

Figure 3.4(a) shows the variation of intra-category and inter-category OBSM similarities for the data set described in Table 3.1. Figure 3.4(b) shows that the corresponding ratio of intra-category over inter-category OBSM similarity remains a non decreasing function of the intra-category overlap. As the intra-cluster overlap increases, more keywords from a common category are shared among documents of a cluster. Therefore, the intra-cluster average similarity increases. Similarly, less keywords from the same category are shared among documents from different clusters; therefore, the

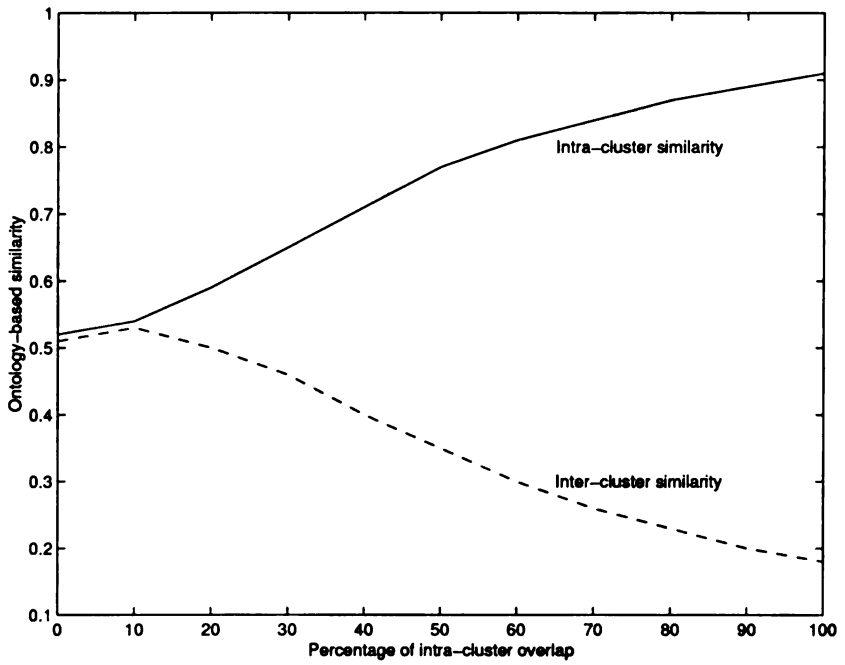


(a) Intra and inter-cluster similarities

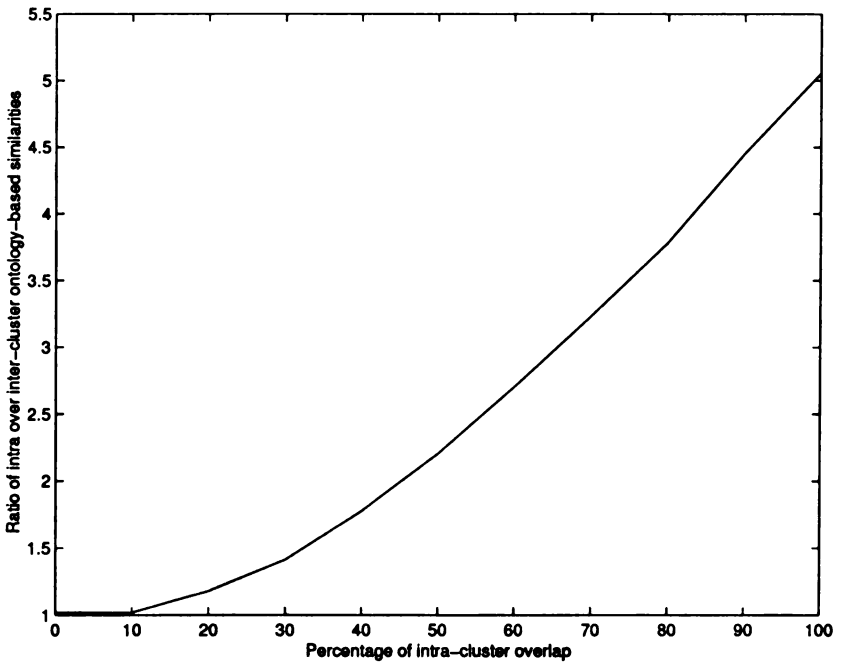


(b) Intra over inter-cluster similarity ratio

Figure 3.3: Intra and inter-cluster OBSM similarities



(a) Intra and inter-cluster similarities



(b) Intra over inter-cluster similarity ratio

Figure 3.4: Intra and inter-cluster OBSM similarities

Dimension (keywords)	20
Number of categories	2
Number of documents	500

Table 3.2: Data set used to generate Figure 3.5

inter-cluster average similarity decreases, resulting in an increasing intra over inter-cluster average similarity ratio.

3.4.2 Varying Inter-cluster Overlap

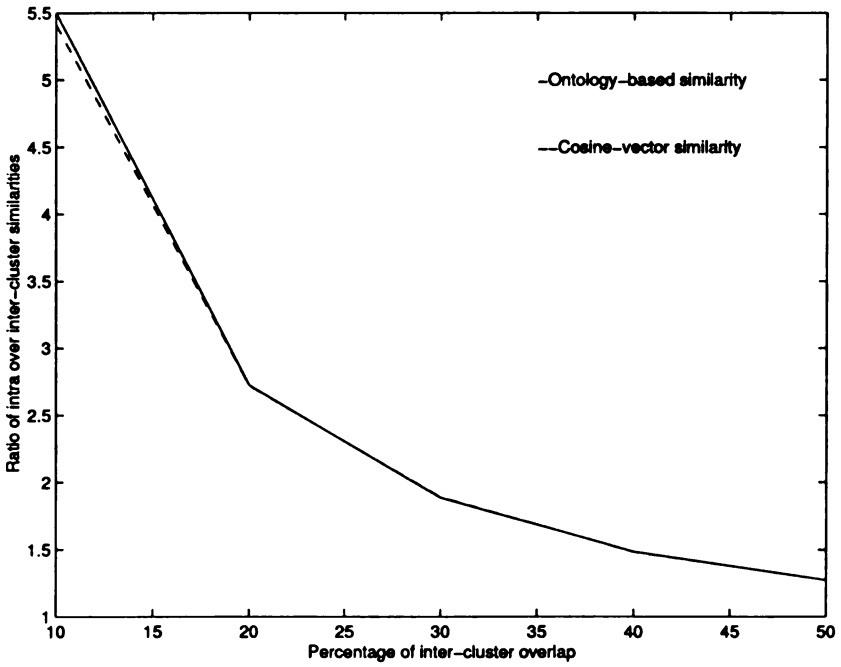


Figure 3.5: Intra over inter-cluster similarity ratio

Figure 3.5 generated for the data set listed in Table 3.2 shows that for small inter-cluster overlap, OBSM yields a slightly larger ratio of intra over inter-cluster similarity than CVSM. Again this is desirable, since a small inter-cluster overlap means better cluster isolation.

3.4.3 Discussion

In both generated data sets, we could see the improvement brought about by combining ontology-based information with keyword-based similarity measure. This improvement was quantified by the isolation and compactness of output clusters of data with varying clustered structure.

3.5 Real Data Set

It should be emphasized that this performance evaluation approach is completely different from the previous one. In fact, in this approach, we collect real documents with no a priori knowledge in terms of initial cluster structure, then we cluster the documents using a clustering technique, using both similarity measures: CVSM and OBSM. This is different from the first approach, where we did not perform any clustering. Instead, we generated clustered data, where the clustered structure is measured by overlap in terms of common keywords. Then, we computed the dissimilarity between these clusters in terms of both similarity measure (standard cosine vector and our proposed similarity measure), and we checked if and how well the cluster structure was preserved.

The methods of web document gathering, pre-processing, and clustering technique are covered in Sections 3.5.1, 3.5.2, and 3.5.3. In section 3.5.4, we address the issue of which web documents do we cluster: a randomly selected subset, or a subset selected from an ontology, like Yahoo [18]. In Section 3.5.4, documents are used from Yahoo's medication index pages.

3.5.1 Web Document Gathering

We needed to automatically gather web documents. We have implemented a gathering utility, based on the "HtDig" robot [31] and "Efficacy" [36], a data gathering system for the World Wide Web, for which I participated in the design and implementation. Our gathering utility starts with a set of initial web sites. Then the robot follows all hyper-links to a level depth specified as a parameter. A cache copy of all documents is created, and an index of web page URLs and all document information is created.

3.5.2 Document Preprocessing

We needed to prepare the web documents for analysis. In the preprocessing phase, we performed the following operations: lexical analysis, elimination of stop words, stemming, construction of term categorization structures.

In lexical analysis, we extracted a stream of selected words from a stream of characters. To do this, we have implemented an automata based lexical analyzer, which scans and identifies the words of a document, removing punctuation marks, and single letters.

We eliminate those words that occur in most documents, called stop words. These words are considered useless because they do not have much discrimination property. This reduces the size of the document database considerably.

We have implemented a stemmer that reduces a word to its origin, or stem, by removing its affixes. For this we have implemented a finite state machine that is based on linguistics rules. The advantage of this operation is that it further reduces

the size of the document database.

We create an ontology in the form of classified hierarchies. It consists of a compiled list of important words in a given domain of knowledge. For each word in the list, a set of related words is defined. A term in the thesauri is usually a single word. Using an ontology in data retrieval allows for adding concepts to words, which significantly increases the quality of clustering.

We used the vector model to represent documents. For term weighting, we used the tf-idf scheme [69]. We use the inverse document frequency because terms that appear in many documents are not very useful for distinguishing documents.

3.5.3 Clustering Technique

The categorization in our clustering method uses a hierarchical clustering, and uses a threshold on the number of steps after which clusters are formed. The main clustering methods considered are: (i) complete link: maximum distance, (ii) single link: minimum distance, and (iii) average link: average distance.

Categories	Document IDs
Business & Economy (Jobs)	1-8
Education	9-18
Government (Military)	19-29
Regional	30-35
Science	36-44
Health	45-51
Recreation & Sports (Sports)	52-63
Entertainment (Movies)	64-76
Computers & Internet	77-82
Total: 9 categories	82

Table 3.3: Real Documents Data Set

Figure 3.6: Hierarchical single link clustering

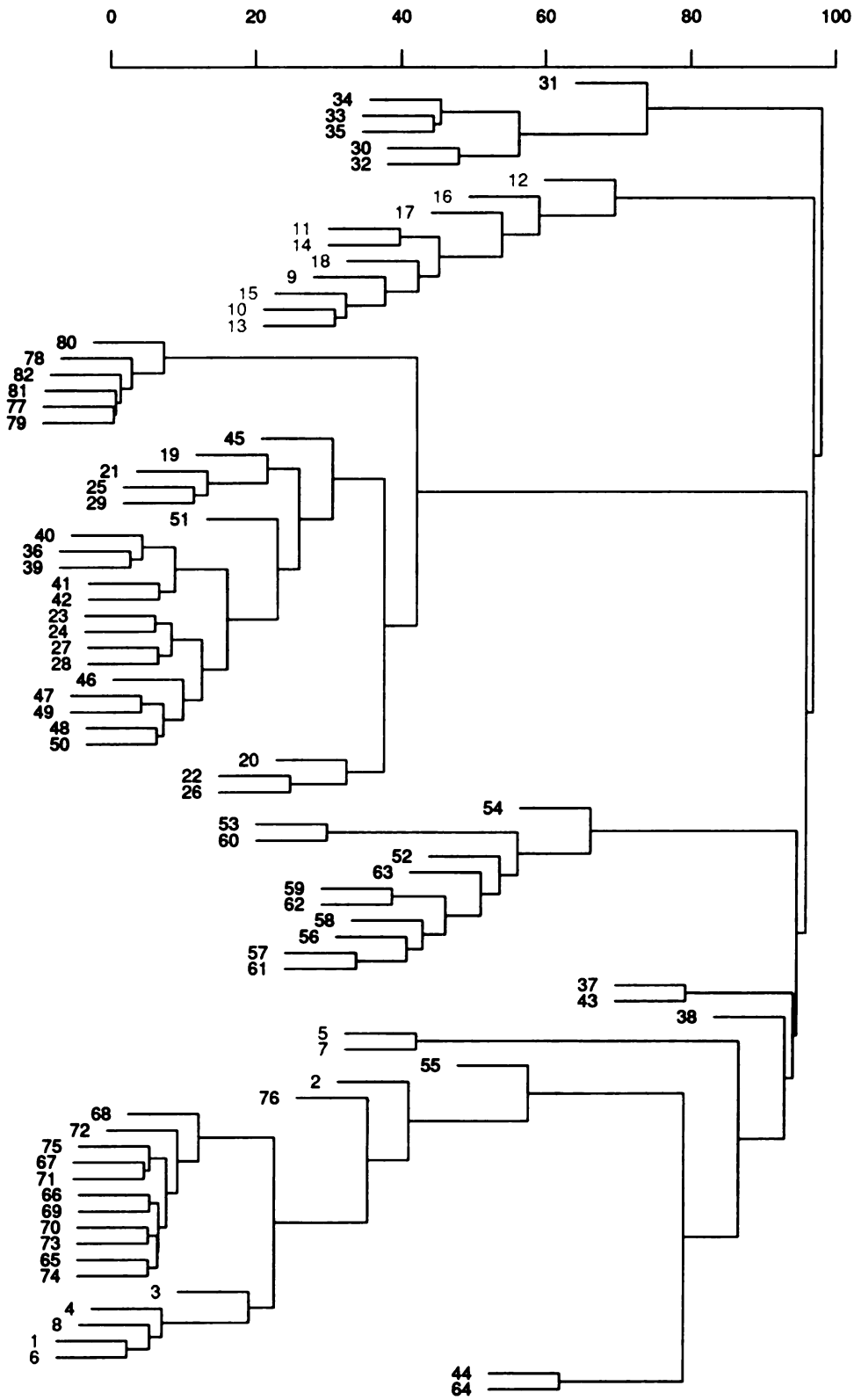


Figure 3.7: Hierarchical average link clustering

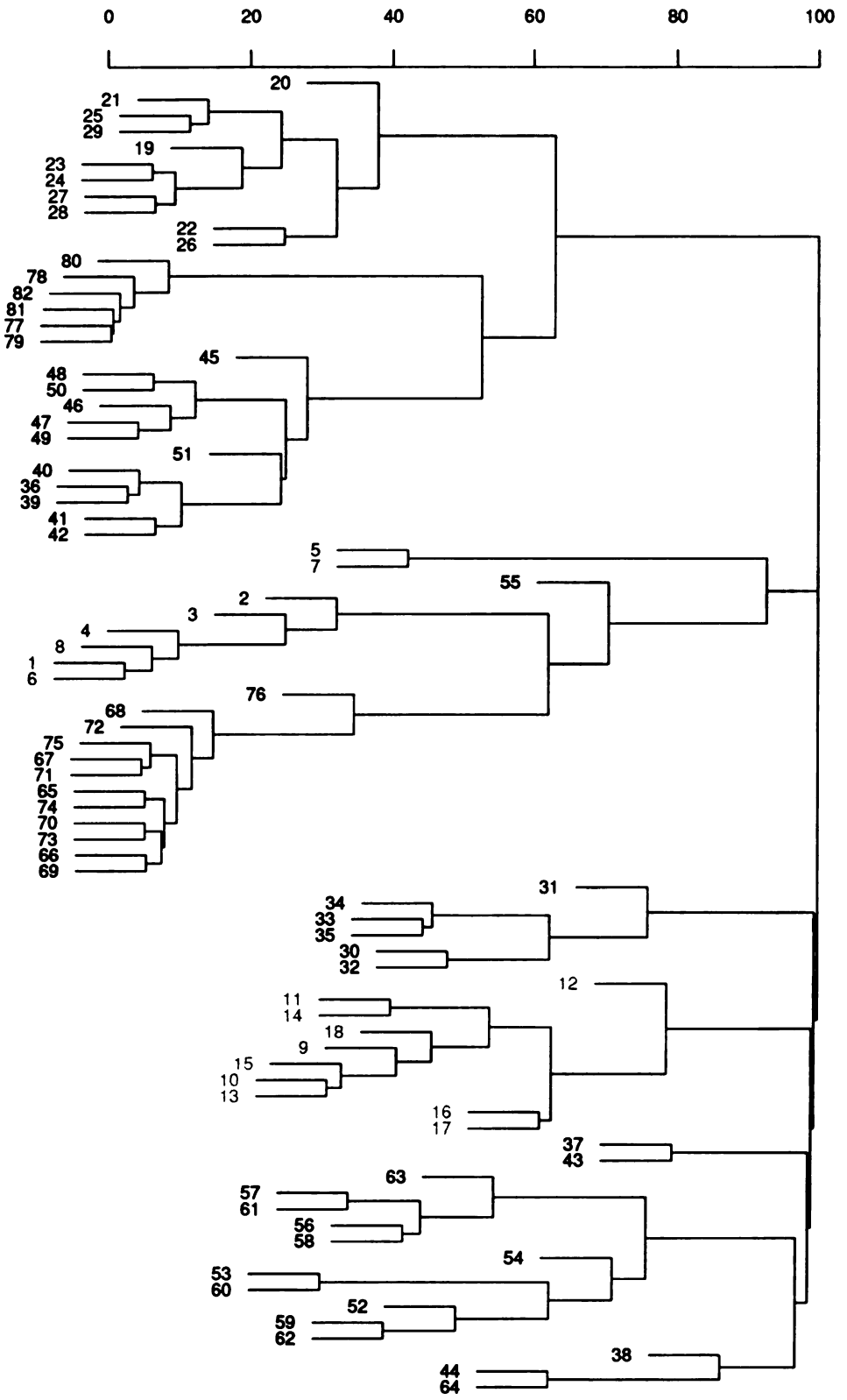


Figure 3.8: Hierarchical complete Link clustering

Figures 3.6, 3.7, and 3.8 show the clustering results of single link, average link and complete link clustering methods for a set of data of 82 documents, taken from nine categories of Yahoo web directory, and listed in Table 3.3. We chose the "complete link" method because it is a conservative method; all pairs of documents must be related before the documents are clustered. In the "single link" method, only a single edge between two large clusters is needed to merge the clusters. Clusters easily chain together and are less compact.

3.5.4 Web Documents Data Set

Now that we decided on the clustering technique, we need to decide on the type of document data set that we need to use for the evaluation. There is a huge number of documents on the web. It is estimated that the Web contains about 320 million pages of information. Most search engines cover only a limited subset of the web. For instance, Lycos [50] covers only about 3 percent of these. Therefore, there are several approaches in deciding which documents to cluster: (i) cluster all web pages gathered following all links, (ii) start from a predefined set of web pages, or (iii) select a subset of web pages within a site (assuming web pages are grouped per site,) for example large pages, or virtual representative pages per site. In order to provide a good evaluation of the practicality and usefulness of OBSM, we used documents from Yahoo's medication index [18].

Description of Data Set Used

In this experiment, We used documents from Yahoo's drugs and medication index[1]. We show the results of the clustering using hierarchy dendrograms, where the documents of a targeted category (Detrol pages) were labeled. A good clustering would be indicated by all labeled documents being grouped together in the same compact cluster, isolated from other clusters.

From Yahoo categories directory, we traced a path from the root of the Yahoo hierarchy to the sub-category "Specific Drugs and Medications". As Figure A.1 of Appendix A shows, this path is: "Home > Health > Pharmacy > Drugs and Medication" where there is a link to "Yahoo! Health: Drugs and Medications" web page, which has links to over 1000 drug and medication pages, listed by alphabetical order. For example under the "D" index[2], a link to Yahoo's Detrol[3] page can be found. That page, for example, gives a description of Tolterodine (brand name: Detrol) "helps patients with an overactive bladder..." Figures A.2 and A.3 of Appendix A show a part of Yahoo's "D" index and Detrol pages.

We used our web gathering utility to automatically download all 2,070 documents from the Yahoo medication index, grouped by drug or type of medication. Then we mixed in all 124 documents that we downloaded from the Detrol[4] web site. We preprocessed the documents using the techniques described in Section3.5.2. The number of distinct keywords in the whole document collection, which represent the

[1]http://health.yahoo.com/health/pdr_drugs/a.html
[2]http://health.yahoo.com/health/pdr_drugs/d.html
[3]http://health.yahoo.com/health/pdr_drugs/0908/0.html
[4]http://www.detrol.com/

dimension of the data set is 9,397. Table 3.4 shows a summary of the data set built.

	Yahoo's medications index	Detrol.com	Total
Number of documents	2,070	124	2,194
Dimension (keywords)	-	-	9,397

Table 3.4: Summary of the documents gathered from Yahoo and Detrol web sites

Clustering Results

We used both CVSM and OBSM to cluster this data set. The resulting clustering dendrograms are shown in Figures 3.9, 3.10 and 3.11. To make the documents of the targeted cluster, i.e. Detrol related documents, easy to spot on the dendrograms, we labeled them by "*" and "+++":

- each "*" represents one of the 124 Detrol.com web site documents.
- each "+++" represent one of the two Detrol related documents contained in the Yahoo medications database.

A good clustering would be indicated by all labeled documents being grouped together in the same compact cluster, isolated from other clusters.

Figure 3.9 shows that with CVSM, the Detrol.com documents did cluster together, but did so in a different cluster than Yahoo's Detrol related documents. This may be because Yahoo's documents share many keywords that are specific to Yahoo's style, such as the type of headers. The effect of these keywords seems to overtake Detrol related keywords.

The clustering dendrograms obtained using OBSM are displayed in Figures 3.10 and 3.11. By using an ontology, we emphasize keywords that are specific to a cate-

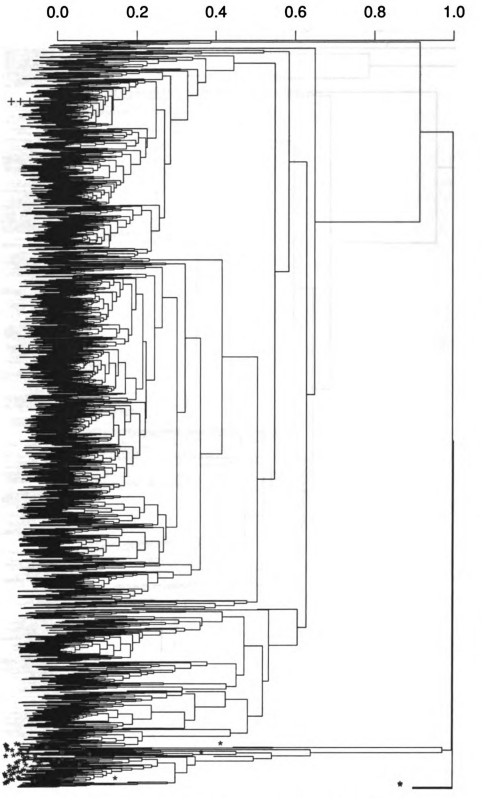


Figure 3.9: Clustering using CVSM

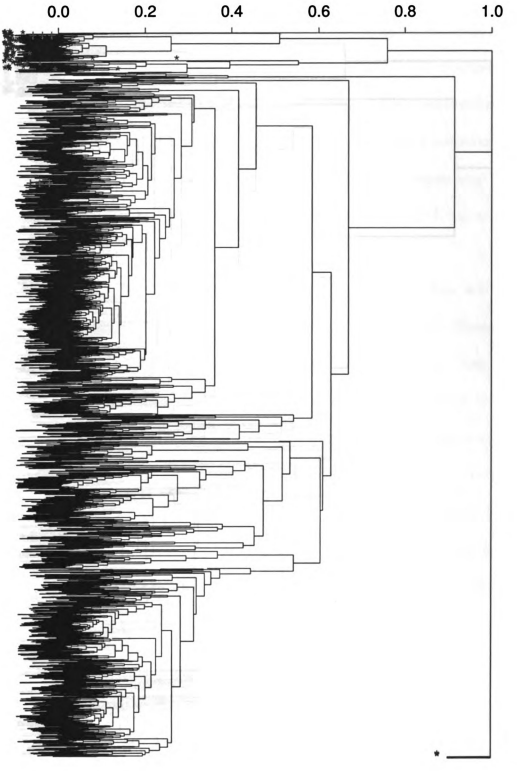


Figure 3.10: Clustering using OBSM (weight 5)

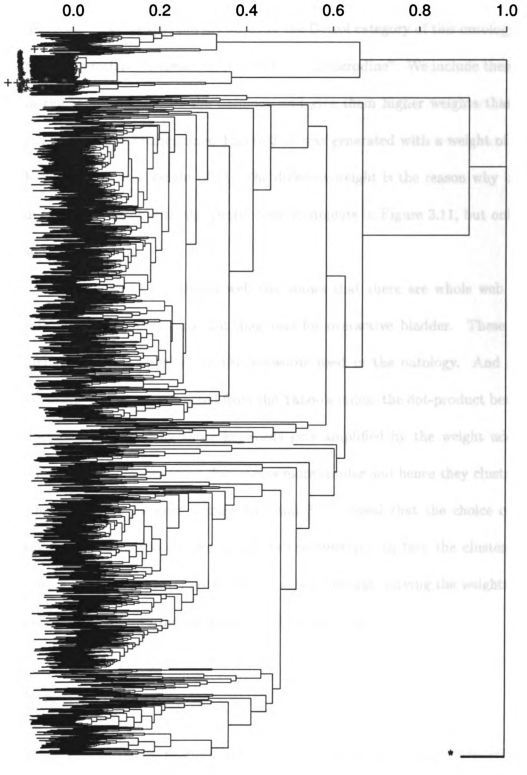


Figure 3.11: Clustering using OBSM (weight 15)

gory that an expert would recognize as relevant to Detrol medication, treatment and diagnosis. Specifically, the keywords in the Detrol category of this ontology are: "Detrol", "Detrolla", "overactive", "bladder", "Tolterodine". We include these keywords in the same category of the ontology and give them higher weights than keywords related to other medications. Figure 3.10 was generated with a weight of 5, whereas Figure 3.11 uses a weight of 15. The different weight is the reason why both Yahoo documents cluster with the Detrol.com documents in Figure 3.11, but only one does so in Figure 3.10.

An analysis of the Detrol web site shows that there are whole web pages that talk about the symptoms and diagnosis for overactive bladder. These web pages include multiple citations of the keywords used in the ontology. And since those keywords have been extracted from the Yahoo's index, the dot-product between those documents induced by those keywords gets amplified by the weight added by the ontology. This makes those documents more similar and hence they cluster together. The differences between Figure 3.10 and 3.11, reveal that the choice of weight of the keywords in the categories affects the clusters. In fact the clusters shown in Figure 3.10 and 3.11 were obtained by experimentally varying the weights between 5 and 20, and monitoring the resulting cluster quality.

3.6 Summary

The results presented in this chapter show that unlike the standard similarity measure, OBSM with a well chosen input ontology for a specific data set consistently reflects

the quality of the clustered structure of the data set. As shown for the data set collected from the Yahoo web site, we could see the improvement brought about by combining ontology-based information with keyword-based similarity measure. This improvement was emphasized in terms of the quality of the cluster of Detrol related web pages. OBSM can improve the quality of clusters, if it is combined with a good choice of domain specific categories.

Chapter 4

Synthetic Document Data Modeling and Generation

4.1 Introduction

This chapter covers synthetic data generation for modeling documents. We investigate data generation in both standard Cartesian coordinate system and the (hyperspherical) polar coordinate system. This investigation includes an analysis of the probability distribution of both Euclidean distance and angle between documents and/or centroids, as well as their variation with respect to the dimension of the data set.

In Section 4.2, we describe our approach, including the parameters and tools used for data analysis. In Section 4.3, we investigate data generation using Cartesian coordinate system. In Section 4.4, we investigate data generation using polar coordinate system. Finally, we summarize our results in Section 4.5.

4.2 Approach

As seen in Chapter 2, if the total number of distinct keywords in a whole collection of documents is t, then a document d_i is represented by the t-dimensional vector: $\vec{d_i} = (w_{i1}, w_{i2}, \ldots, wit)$. We generate synthetic data for modeling documents by representing a document by a feature vector in the t-dimensional vector space. Document data has some intrinsic structure, and document points are generally scattered around cluster centroids. A cluster centroid may be viewed as a representative point of the cluster. Synthetic data generation is controlled by the following parameters:

1. the total number of documents;
2. the data set dimension, i.e. the number of keywords;
3. the number of clusters, or categories;
4. the size (number of documents) of clusters;
5. the centers of clusters, or centroids;
6. the average scatter around cluster centroids, or cluster compactness;
7. the inter-cluster overlap, or cluster isolation.

The average scatter around a cluster centroid measures the cluster compactness, and is defined by the interval within which data points are scattered around the cluster centroid. This is sometimes represented by the cluster radius [82]. The inter-cluster isolation indicates how the clusters are isolated from each other, and may be defined

by the average number of documents that fall closer to a cluster centroid other than their own. In the following section, we formally define the above parameters.

4.2.1 Definitions

In the following, we define the centroid and radius of a cluster, as well as the intra-cluster and inter-cluster average distances. Let us assume that we have a data set with the following parameters: (i) the dimension is t, (ii) the total number of points is N, (iii) the number of clusters is K, (iv) the number of documents per cluster C_k is n_{ck}. If a cluster has n_{ck} t-dimensional data points $\{\vec{d_i}\}$ $(i = 1, 2, ..., n_{ck})$, then the centroid $\vec{C}$, radius R, the intra-cluster and inter-cluster average distances are defined respectively as follows:

$$\vec{C} = \frac{1}{n_c}\sum_{i=1}^{n_c}\vec{d_i} \tag{4.1}$$

$$R = (\frac{1}{N}\sum_{i=1}^{n_c}(\vec{d_i} - \vec{C})^2)^{\frac{1}{2}} \tag{4.2}$$

$$d_c = (\frac{1}{n_c(n_c-1)}\sum_{i=1}^{n_c}\sum_{j=1}^{n_c}(\vec{d_i} - \vec{d_j})^2)^{\frac{1}{2}} \tag{4.3}$$

$$d_{12} = (\frac{1}{n_{c1}n_{c2}}\sum_{i=1}^{n_{c1}}\sum_{j=1}^{n_{c2}}(\vec{d_{1i}} - \vec{d_{2j}})^2)^{\frac{1}{2}} \tag{4.4}$$

4.2.2 Distribution Analysis of Random Variables

In general, the distribution of a random variable is represented by two (related) functions. The probability distribution function (PDF) of a random variable X is

$$F_X(x) = P[X \leq x] \tag{4.5}$$

One main property of the PDF is that $F_X(x)$ is a nondecreasing function of x, with range $[0, 1]$.

The probability density function (pdf) of a random variable X is

$$f_X(x) = \frac{dF_X(x)}{dx} \tag{4.6}$$

One main property of the pdf is that

$$\int_{-\infty}^{\infty} f_X(x)\, dx = 1 \tag{4.7}$$

The two distributions used in this chapter are the uniform (U) and normal (N) distributions, with probability density functions:

$$f_U(x) = \frac{1}{b-a} \quad \text{for} \quad a < x < b \tag{4.8}$$

$$f_N(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \quad exp \quad (-\frac{1}{2}[\frac{x-\mu}{\sigma}]^2) \tag{4.9}$$

Normal distributions are characterized by the mean σ and standard deviation μ. They all share a common property, the 68-95-99.7% Empirical Rule: 68% of the observations fall within 1 deviation of the mean; 95% of them fall within 2 deviations; and 99.7% fall within 3 deviations.

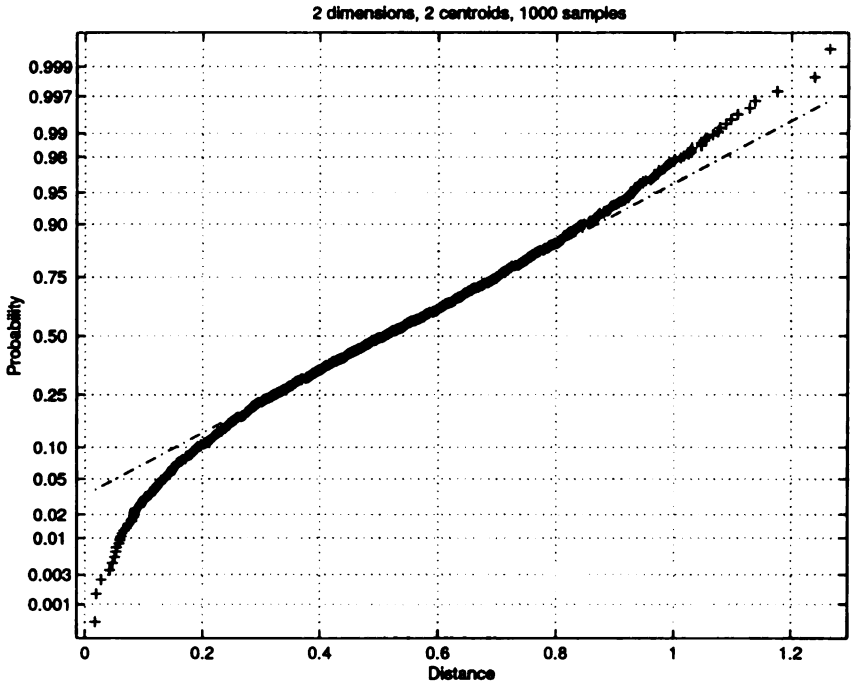
4.3 Cartesian Coordinate System

We first investigated the generation of synthetic data in the standard Cartesian coordinate system. In this vector space, documents are represented by t-dimensional vectors. Every dimension represents an index term or keyword, where t is the total number of distinct index terms in the documents data set. The vector components are the keyword weights for the corresponding document.

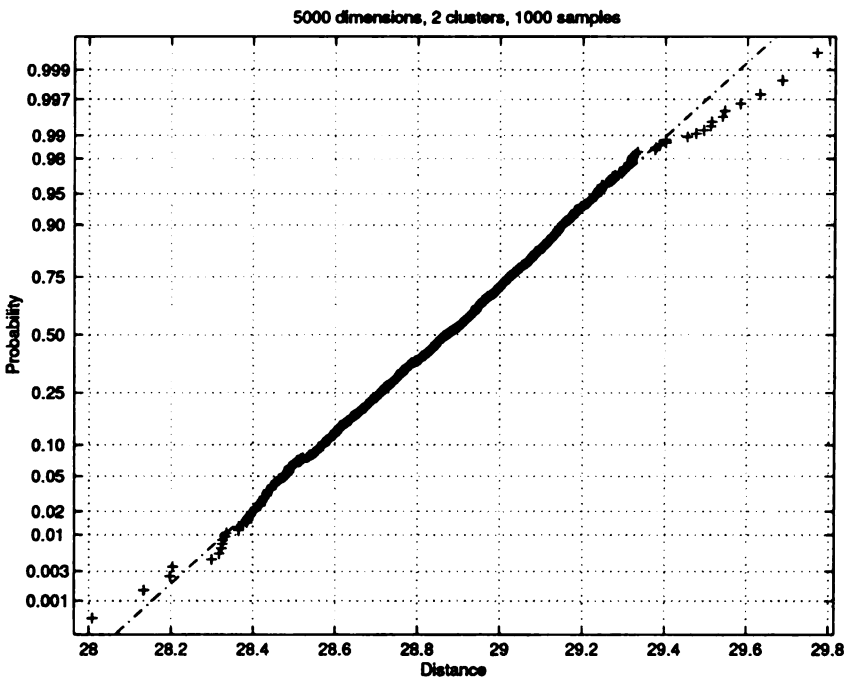
4.3.1 Distribution of Euclidean Distance for Uniform Random Data

We first investigated the distribution of Euclidean distance between two random vectors, representing cluster centroids, for various dimensions. Since we could not assume a priori knowledge about the cluster centroids distribution, we treated the Cartesian coordinates (weights) as a random uniform variable. We run our generation program a 1000 times with various seeds, and monitored the distance distribution.

Figures 4.1(a) and (b) show the distribution of Euclidean distance between two cluster centroids for dimension=2 and 5000 respectively. In the case of 2 dimensions, it has an average of .516 and standard deviation of 0.25; the plot shows that it is hardly a normal distribution. However, in the case of 5000 dimensions, the plot shows that it is a normal distribution with average 28.87 and standard deviation 0.23.



(a) Dimension=2



(b) Dimension=5000

Figure 4.1: Distribution of distance between centroids

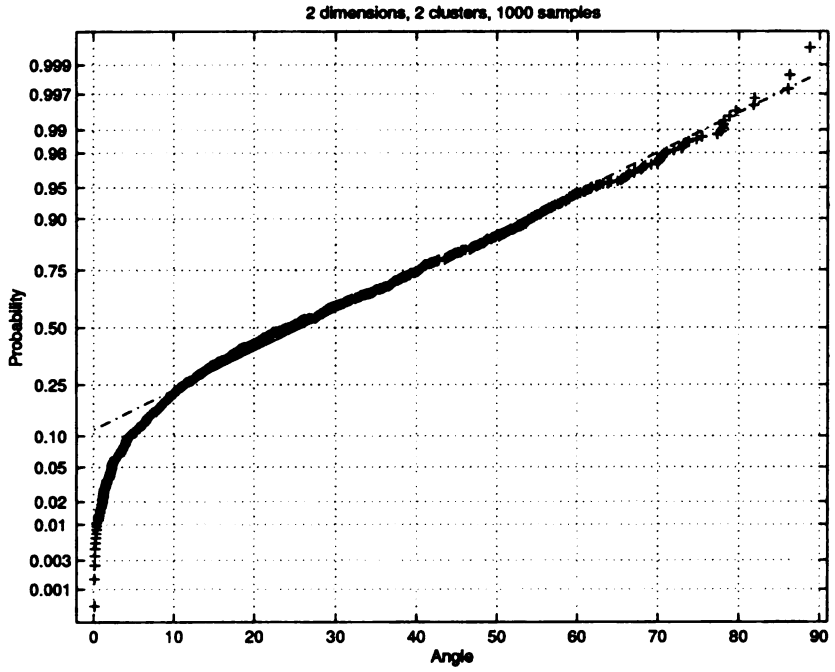
4.3.2 Distribution of Angle for Uniformly Distributed Random Data

Using the same data set as in the previous section, we investigated the distribution of the angle between documents or cluster centroids, for various dimensions. Other studies [63, 65] have suggested some properties of angle in high dimension in the context of nearest neighbor queries.

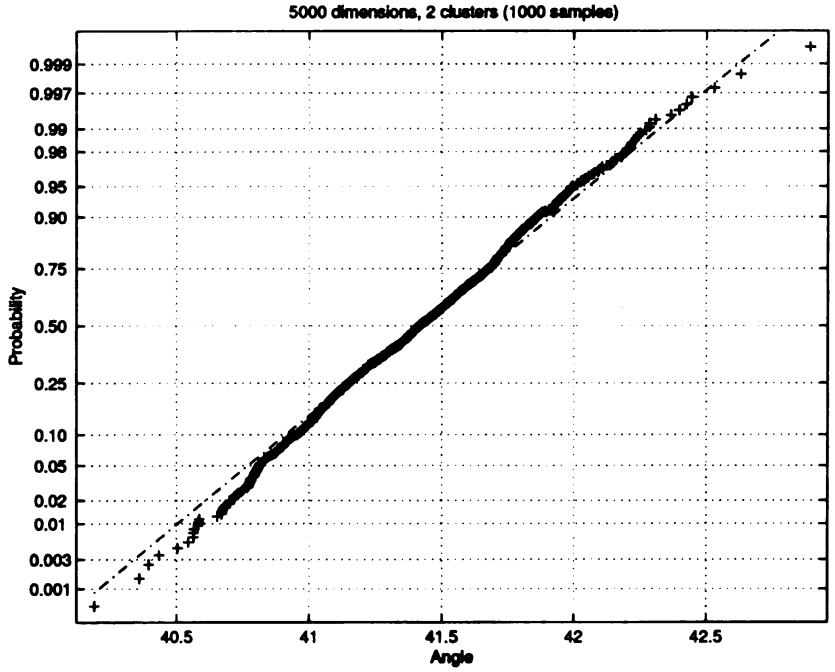
Figures 4.2(a) and (b) show the distribution of angle between centroids for the same data set as Figures 4.1(a) and (b). In the case of 2 dimensions, it has an average of 27.3 and standard deviation of 19.0; the plot shows that it is hardly a normal distribution. However, in the case of 5000 dimensions, the plot shows that it is a normal distribution with average 41.4 and standard deviation 0.36.

4.3.3 Average Distance and Angle, and Deviation as Functions of Dimension

We have extended the previous experiments for multiple data sets, having various dimensions. Figures 4.3(a) and (b) show the plot of average Euclidean distance and angle between two cluster centroids versus dimension (number of terms); the components of the centroids being randomly scattered over the interval $[0, 1]$ following a uniform distribution. While the Euclidean distance fits the $\sqrt{\frac{dim}{6}}$ function, the angle converges asymptotically towards 45 degrees as the dimension goes to infinity. Those figures also show the standard deviation as a function of dimension. The data plotted in those figures is described in Tables 4.1 and 4.2. Table 4.1 shows the distance values

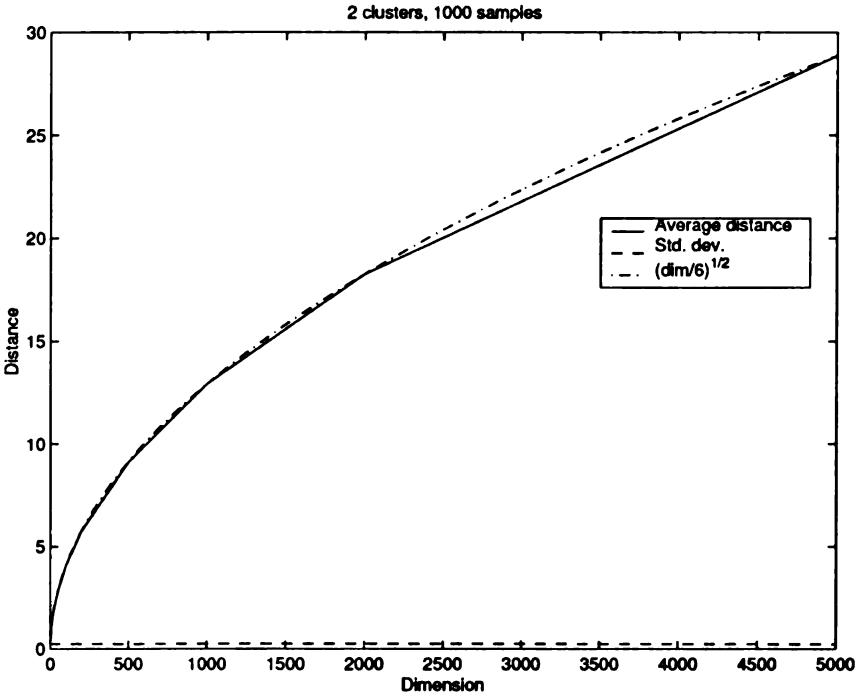


(a) Dimension=2

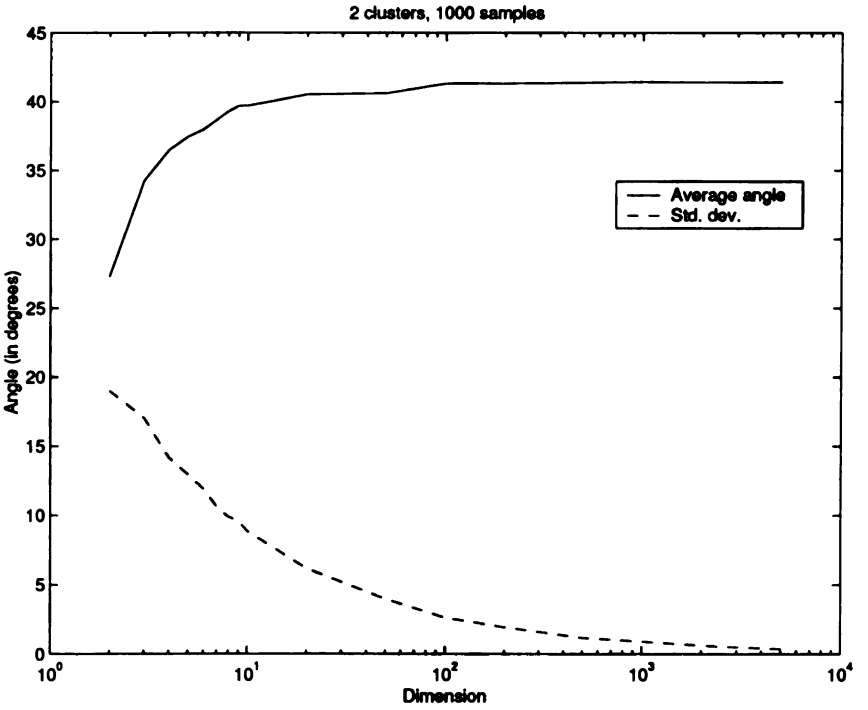


(b) Dimension=5000

Figure 4.2: Distribution of angle between centroids



(a) Euclidean distance



(b) Angle distance

Figure 4.3: Average distance between cluster centroids

Dimension	Euclidean distance	Deviation
2	0.5165	0.2455
3	0.6616	0.2447
4	0.7787	0.2446
5	0.8779	0.2528
6	0.9655	0.2499
7	1.0490	0.2461
8	1.1299	0.2478
9	1.2037	0.2552
10	1.2673	0.2478
20	1.8054	0.2427
50	2.8429	0.2444
100	4.0802	0.2331
200	5.7660	0.2383
500	9.1180	0.2354
1000	12.9166	0.2510
2000	18.2539	0.2479
5000	28.8672	0.2335

Table 4.1: Distance between cluster centroids (2 clusters, 1000 samples)

Dimension	Angle	Deviation
2	27.2858	19.0131
3	34.2791	17.0340
4	36.4735	14.1604
5	37.4477	12.9496
6	37.9835	11.9038
7	38.6798	10.6275
8	39.2921	9.9100
9	39.6707	9.5699
10	39.6838	8.8548
20	40.5166	6.1592
50	40.6047	3.9594
100	41.3060	2.6153
200	41.3147	1.9324
500	41.3683	1.1768
1000	41.4426	0.8923
2000	41.4035	0.6100
5000	41.4151	0.3666

Table 4.2: Angle between cluster centroids (2 clusters, 1000 samples)

Dimension	1000
Number of clusters	20
Number of documents	2000
Radius	2

Table 4.3: Summary of data set used to analyze intra and inter-cluster distance

for dimension up to 5000, for the case of two clusters, with 1000 samples. Table 4.2 shows the angle values for dimension up to 5000, for the case of two clusters, with 1000 samples.

4.3.4 Distribution of Intra and Inter-Cluster Distances

We then investigated the distribution of intra and inter-cluster distance or clustered data.

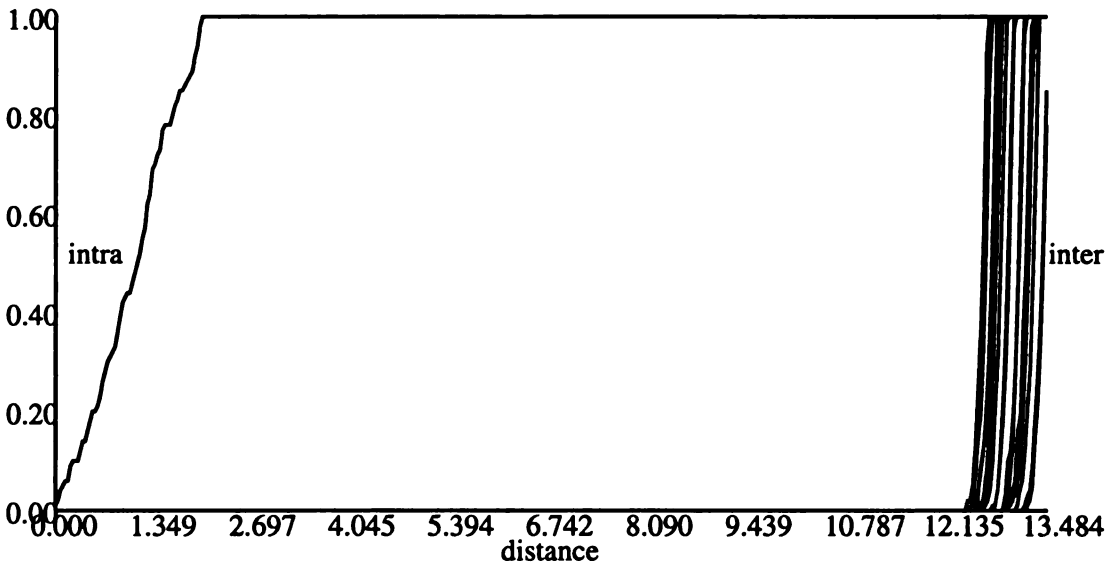


Figure 4.4: PDF of intra and inter-cluster distances

For the data set summarized in Table 4.3, Figure 4.4 shows the PDF of intra-cluster distance for cluster 1, and the PDFs of inter-cluster distances between cluster 1 and all other clusters. Table 4.4 shows the corresponding averages and deviations. The

Cluster	Distance from centroid 1	Average intra cluster distance	Deviation
1	0	1.0541	0.548814
		Average inter cluster distance	
2	13.117636	13.0601	0.060046
3	12.804267	12.7554	0.048568
4	12.985431	12.9205	0.056116
5	12.758499	12.6982	0.061719
6	13.141036	13.0786	0.058101
7	12.809853	12.7614	0.050714
8	13.454486	13.3826	0.068648
9	13.249435	13.1818	0.064184
10	12.640610	12.5795	0.057982
11	12.824699	12.7610	0.059478
12	12.891011	12.8274	0.059240
13	13.233682	13.1606	0.066985
14	12.674950	12.6179	0.059596
15	13.328342	13.2672	0.063191
16	12.795399	12.7443	0.049922
17	13.467176	13.3955	0.064418
18	12.760791	12.6972	0.061297
19	12.996210	12.9340	0.059926
20	13.254703	13.1969	0.063341

Table 4.4: Intra and inter-cluster average distance and deviation

first row of Table 4.4 shows the intra-cluster average distance and deviation for cluster 1. The remaining rows show the distances between cluster 1 centroid and all other centroids, as well as the inter-cluster average distances, and their deviations. The fact that all the inter-cluster average distances are all in the range of 12-13, which is much greater than the intra-cluster average distance (1.054) indicates that cluster one is well isolated from other clusters. The standard deviation (essentially the steepness of the PDF around the average) characterizes the compactness of cluster 1.

4.3.5 Expected Value of Eucliden Distance

In this section, we derive the expected value of the Euclidean distance between two points in t-dimensional space, with components uniformly distributed over interval [0,1].

Let's suppose that X_i and Y_i are two independent random variables, uniformly distributed over the interval [0,1]. The probability distribution function (PDF) of X_i is $F_{X_i}(x) = x$, for $0 < x <= 1$ (same for F_{Y_i}). The probability density function (pdf) of X_i is $f_{X_i}(x) = \frac{dF_{X_i}(x)}{dx} = 1$, for $0 < x < 1$ (same for f_{Y_i}). Then $f_{X_iY_i}(x, y) = f_{X_i}(x)f_{Y_i}(y) = 1$, for $0 < x, y < 1$

$$
\begin{aligned}
E(dis(X,Y)) &= E\Big(\big(\sum_{i=1}^{t}(X_i - Y_i)^2\big)^{\frac{1}{2}}\Big) \\
&= \Big(\sum_{i=1}^{t}\int_0^1\int_0^1 (X_i - Y_i)^2 f_{X_iY_i}(x,y)\,dx\,dy\Big)^{\frac{1}{2}} \\
&= \Big(\sum_{i=1}^{t}\int_0^1\int_0^1 (X_i - Y_i)^2\,dx\,dy\Big)^{\frac{1}{2}} \\
&= \big(\frac{t}{6}\big)^{\frac{1}{2}}
\end{aligned}
\tag{4.10}
$$

4.3.6 Distribution of Intra-Cluster Angles

For a cluster of documents scattered uniformly around the cluster centroid, we determine the distribution of intra-cluster angles.

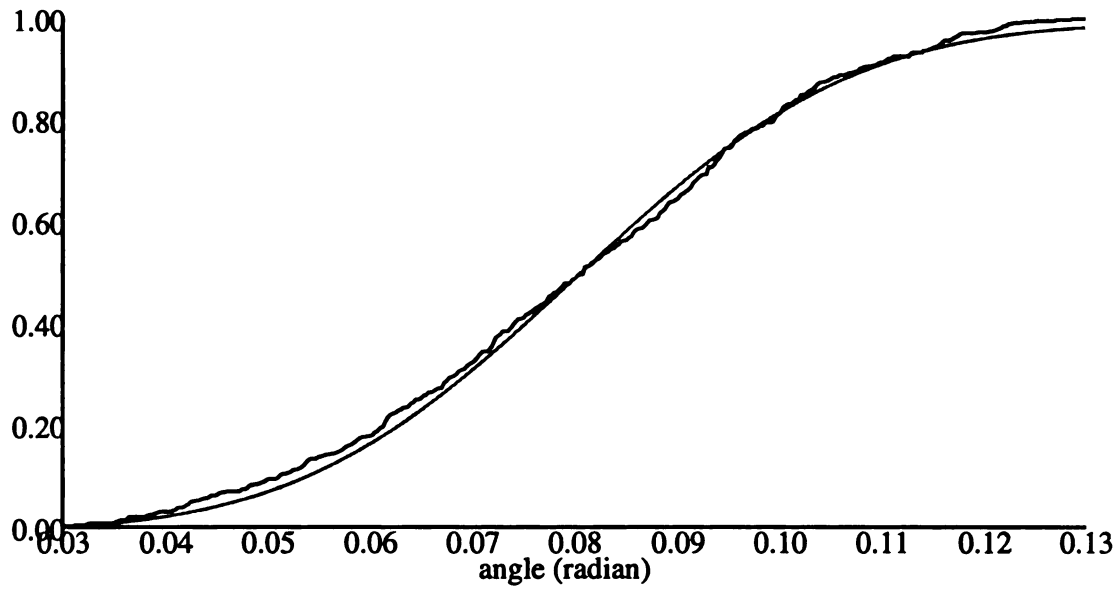


Figure 4.5: PDF of intra-cluster angles

Fig 4.5 shows the probability distribution function (dark line) of the intra-cluster angle, for a data set with 1000 keywords, 2 clusters, 1000 documents (length=1.00, radius=0.145). It also shows the theoretical normal (Gaussian) distribution function

with avg=0.08 and dev=0.0216 (gray line). The intra-cluster hyper angles are distributed following a normal distribution. This type of distribution is similar to that of the distribution of intra-cluster distance shown in Fig 4.4. This shows that the generation of synthetic data using the standard Cartesian coordinate system yields comparable results in terms of the distribution of Euclidean distance and angle based similarities.

4.4 Polar Coordinate System

Since as we increased dimension the expected angle value for uniformly distributed documents asymptotically converges to a constant, it is not possible to vary document cluster separation. So we decided to use the (hyper-spherical) polar coordinate system instead of the Cartesian coordinate system to generate our synthetic clustered data set. We also decided to use the normal (Gaussian) distribution with various standard deviations to scatter documents within clusters. This allows for using different cluster spread values around centroids to specify different cluster compactness. Because documents belonging to the same cluster have similar angle, it is easier to visualize clusters using polar instead of Euclidean coordinates. In the polar coordinate system, we can specify documents belonging to the same cluster by giving them the same mean angle value. The separation between documents in the same cluster will be controlled by the standard deviation around centroid polar coordinates. We also use angle based overlap to specify cluster isolation. It is worth mentioning that using hyper spherical polar coordinate system to generate our synthetic clustered data set

is intuitively more appropriate for modeling documents since we are using angle based similarity measure.

4.4.1 Polar vs. Cartesian Coordinates in t-dimensional Space

Since documents are characterized by keywords, to represent documents by polar coordinates, we needed to derive the polar angles that represent documents in function of the keywords.

In 2-dimensional space, for $0 \leq r \leq +\infty, 0 \leq \theta \leq 2\pi$:

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} r\cos\theta \\ r\sin\theta \end{pmatrix} \tag{4.11}$$

In 3-dimensional space, for $0 \leq r \leq +\infty, 0 \leq \theta \leq 2\pi, 0 \leq \psi \leq \pi$:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} r\sin\psi\cos\theta \\ r\sin\psi\sin\theta \\ r\cos\psi \end{pmatrix} \tag{4.12}$$

Figure 4.7 shows 3-dimensional (spherical) polar coordinate system. In general, in a t-dimensional system, where $k_1, k_2, \ldots, k_t$ are the dimensions representing the keywords, a document d_i in a data set with t keywords will be represented in t-dimensional space by a point with $t-1$ angles, and the length $|d_i|$. The relationship between the Cartesian coordinates $w_{i1}, w_{i2}, \ldots, w_{it}$ and the $t-1$ angles $\theta_1, \theta_2, \ldots, \theta_{t-1}$ is:

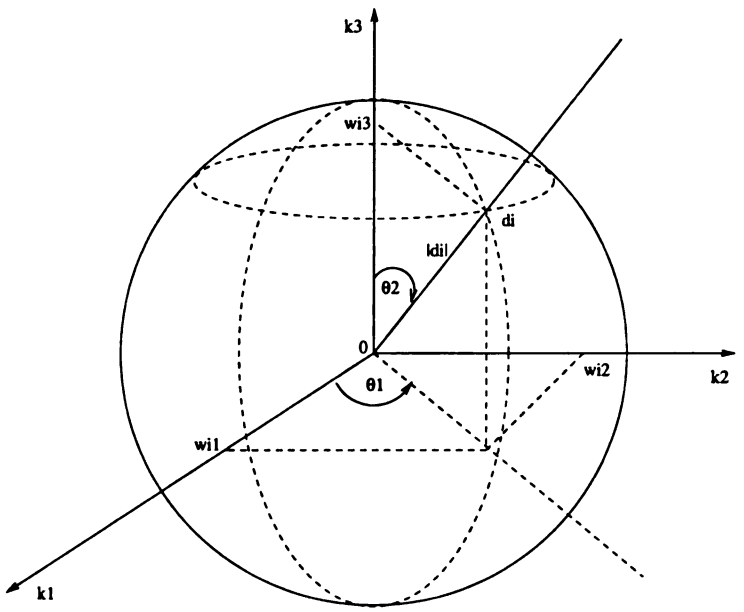


Figure 4.6: Spherical coordinate system

$$\begin{pmatrix} w_{i1} \\ w_{i2} \\ w_{i3} \\ \cdots \\ w_{i\ t-1} \\ w_{it} \end{pmatrix} = \begin{pmatrix} |d_i| \sin\theta_{t-1} \sin\theta_{t-2} \ldots \sin\theta_2 \cos\theta_1 \\ |d_i| \sin\theta_{t-1} \sin\theta_{t-2} \ldots \sin\theta_2 \sin\theta_1 \\ |d_i| \sin\theta_{t-1} \sin\theta_{t-2} \ldots \cos\theta_2 \\ \cdots \\ |d_i| \sin\theta_{t-1} \cos\theta_{t-2} \\ |d_i| \cos\theta_{t-1} \end{pmatrix} \tag{4.13}$$

where $0 \leq \theta_1 \leq \frac{\pi}{2}$ is the horizontal angle measured on the K_1K_2 plane from the K_1 axis, and $0 \leq \theta_2, \theta_3, \ldots, \theta_{t-1} \leq \frac{\pi}{2}$ are the polar angles measured from the $K_3, K_4, \ldots, K_t$ axes.

4.4.2 Clustered Document Data

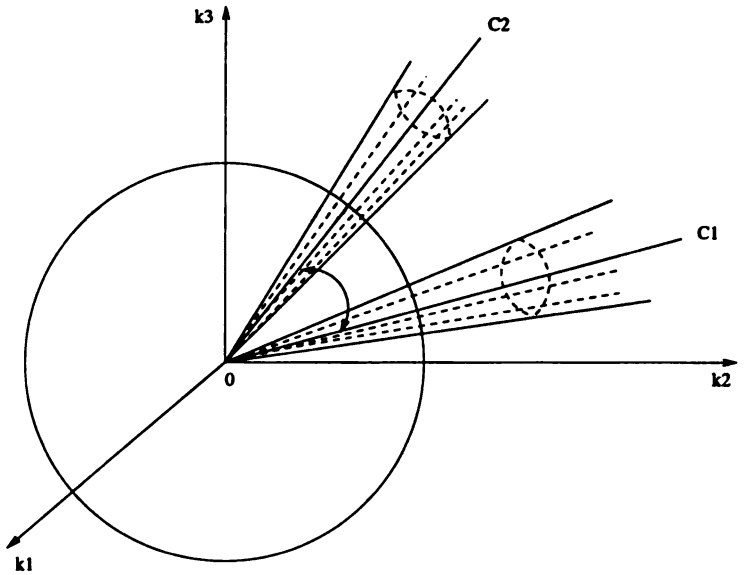


Figure 4.7: Representation of clustered documents in polar coordinate system

Figure 4.6 shows the representation of clustered documents in polar coordinate system. $C_1, C_2, \ldots$ represent the centroids, around which document data is clustered. We generate clustered data around hyper-cones representing centroids, around which, within a delta angle, documents can be generated in the vector space. In the (hyper-spherical) polar coordinate system, the parameters of synthetic data, is represented as follows: (1) the keywords are represented by the dimensions of the vector space; (2) the clusters are represented by angles; (3) the documents are represented by the points in the vector space; (4) the similarity between documents is represented by angles defined from the origin.

Generation of Clustered Document Data

In the polar coordinate system, a random point can be generated as $[r \cos \theta_k] \quad 1 \leq k \leq t$, where r and θ_k are random polar coordinates. θ_k is distributed over $[0, \frac{\pi}{2}]$,

and r and θ_k are independent. We can also generate the point as $[r\cos(\frac{\pi}{2}U_k)]$, where U_k is distributed over $[0, 1]$.

Algorithm The following is an algorithm for generating clustered document data, adapted from [34].

Step 1. Establish cluster sizes

Repeat Step 2 to 4 for all clusters

Step 2. Select a cluster centroid as uniformly distributed random variable,

Step 3. Scatter documents in the hyper-cone surrounding the average centroid angle: Select the t polar coordinates (angles) of each document relative to the cluster centroid as a normal distributed random variable with centroid components as average, and deviation (or spread) σ. If a document falls outside the domain, repeat Step 3.

Step 4. Repeat steps 2 and 3 until all cluster overlap is within maximum allowed.

We generated data sets with different dimensions, and different average scatter angle around centroids.

Figure 4.8 shows an example data set for 1,000 documents, scattered around 5 clusters in 2 dimensions with 5 degrees spread and a maximum 30% overlap allowed. The five clusters are labeled respectively '.', '+', '*', 'x' and 'o'. Table 4.5 shows the overlap (in %) between the five clusters (average=3.25%).

Then we increased the dimension while keeping the same average scatter angle.

Figure 4.9 shows an example data set for 10,000 documents, scattered around 5 clusters in 3 dimensions with 5 degrees spread and a maximum 30% overlap allowed.

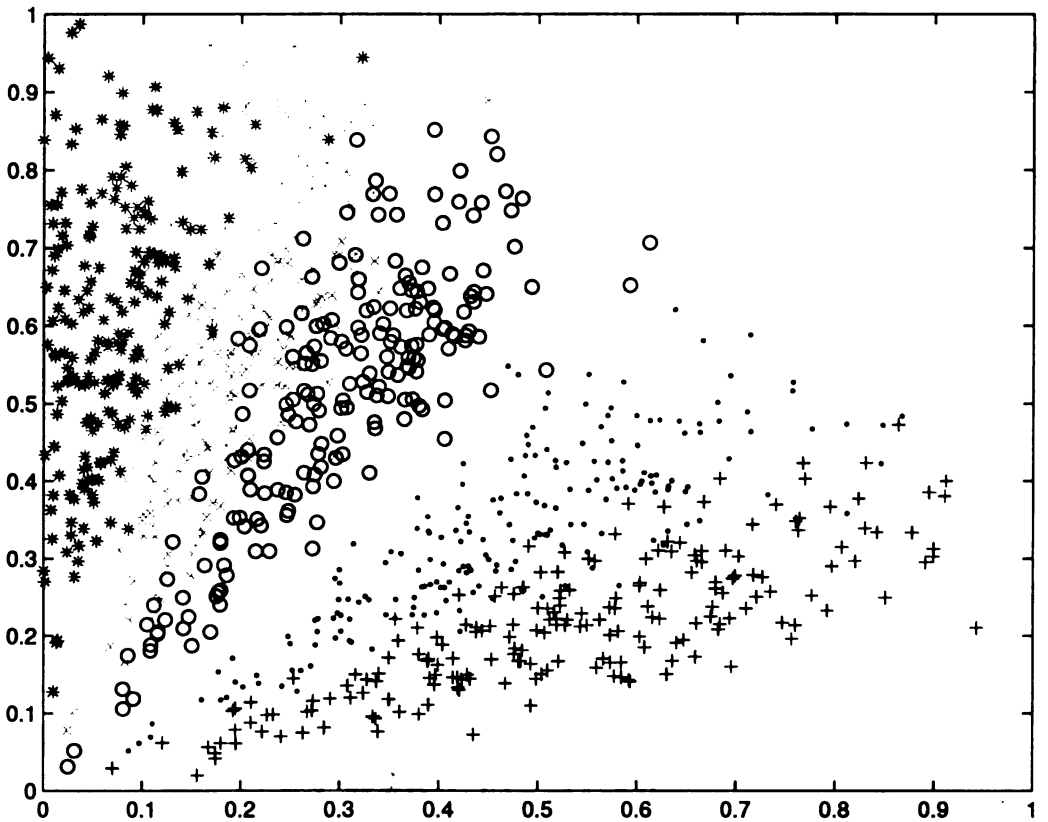


Figure 4.8: Clustered synthetic document data (2 dim, 5 deg. scatter)

	1	2	3	4
2	8.75			
3	0.00	0.00		
4	0.00	0.00	8.50	
5	0.75	0.00	0.75	13.75

Table 4.5: Inter-cluster overlap (2 dim, 5 deg. scatter)

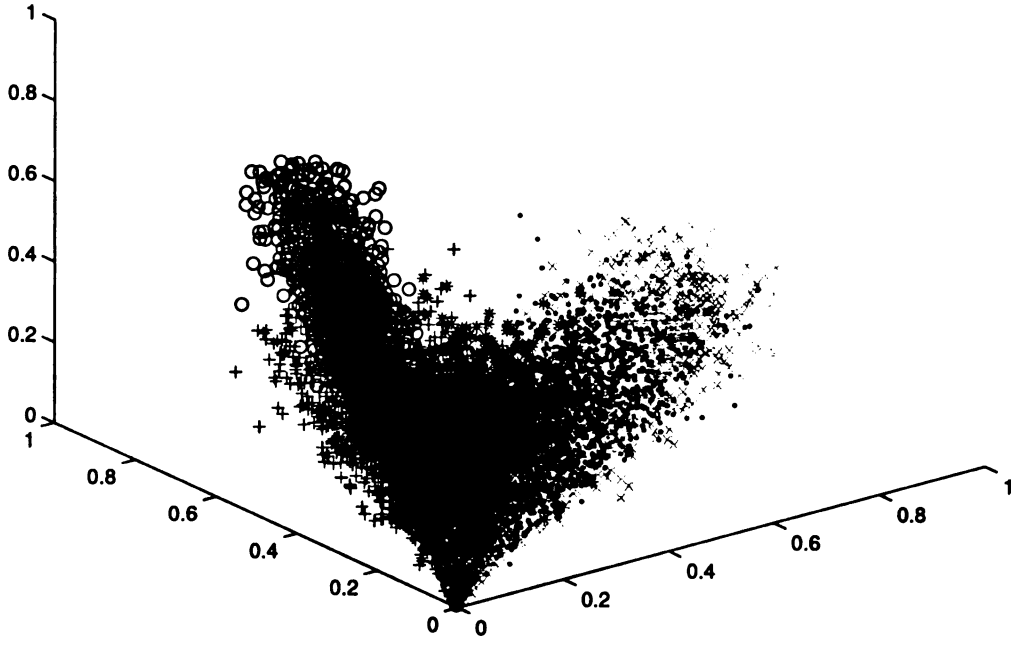


Figure 4.9: Clustered synthetic document data (3 dim, 5 deg. scatter)

Here again, the five clusters are labeled respectively '.', '+', '*', 'x' and 'o'. Table 4.6 shows the overlap (in %) between the five clusters (average=2.53%).

	1	2	3	4
2	0.27			
3	4.67	9.15		
4	11.03	0.03	0.20	
5	0.00	0.00	0.00	0.00

Table 4.6: Inter cluster overlap (3 dim, 5 deg. scatter)

Then we increased the average scatter angle while keeping the same dimension.

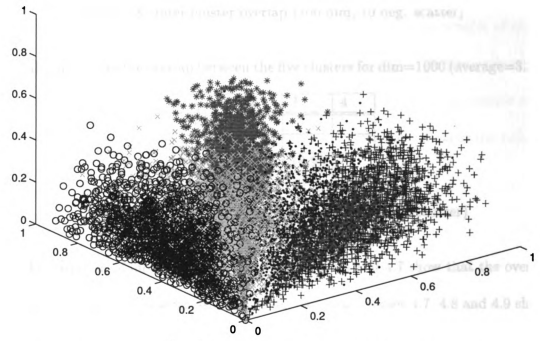


Figure 4.10: Clustered synthetic document data (3 dim, 10 deg. scatter)

Figure 4.10 shows an example data set for 10,000 documents, scattered around 5 clusters in 3 dimensions with 10 degrees spread and a maximum 30% overlap allowed. Table 4.7 shows the overlap (in %) between the five clusters (average=4.36%).

	1	2	3	4
2	9.95			
3	0.33	0.50		
4	9.75	0.55	0.40	
5	1.52	0.00	0.05	20.62

Table 4.7: Inter cluster overlap (3 dim, 10 deg. scatter)

Then we increased the data set dimension further. Table 4.8 shows the overlap between the five clusters for dim=100 (average=4.06%).

	1	2	3	4
2	10.12			
3	0.05	0.00		
4	2.65	3.05	0.00	
5	5.22	5.53	0.00	13.98

Table 4.8: Inter cluster overlap (100 dim, 10 deg. scatter)

Table 4.9 shows the overlap between the five clusters for dim=1000 (average=3.31%).

	1	2	3	4
2	0.85			
3	1.95	0.00		
4	2.35	0.05	16.98	
5	5.65	3.05	1.05	1.20

Table 4.9: Inter cluster overlap (1000 dim, 10 deg. scatter)

The variation of inter-cluster overlap in Tables 4.6 and 4.7 show that the overlap increases with average scatter around cluster centroids. Tables 4.7, 4.8 and 4.9 show, for the same scatter angle, a slight decrease of overlap as dimension increases. Therefore, the inter-cluster overlap can be controlled by the average scatter value around cluster centroids.

4.5 Summary

We first generated synthetic data using the standard Cartesian coordinate system, and showed comparable results in terms of the distribution of Euclidean distance and angle based similarities when using uniform random data. We then used the polar coordinate system to apply normal distribution to model cluster compactness and isolation. This is unlike generating data in the Cartesian coordinate system, where controlling parameters for a desired distribution in terms of angles, is not easily achievable by Cartesian parameters. The data sets that we generated in this chapter show that we can choose data with various parameters. Our data generation mechanism based on polar coordinate system achieves easy control of cluster centroids, cluster compactness and cluster isolation. We specify cluster compactness by using normal distribution where standard deviation controls how to scatter documents around cluster centroids. And we specify cluster isolation by using inter-cluster angle-based overlap.

Chapter 5

Semantic Groups of Graph Data Model and Documents

5.1 Introduction

In previous chapters we have developed clustering algorithms based on ontologies. For web applications, many ontologies may have to be defined and these ontologies are related to each other. The ontologies also evolve as the document database changes and grows. In order to apply the proposed clustering to a database of documents, a framework to represent clusters of documents is needed. For this, a graph data model that uses semantic groups to add semantics to documents and keywords is suggested. Semantic groups can be used in the management of multiple ontologies. They can be built using keywords based on output clusters. These semantic groups can then be used in input ontologies for subsequent clustering. This reuse of semantics incrementally improves the quality of clustering. Moreover, since clustering requires

document processing and a structure to hold post-processed document data, the graph data model can also be used as an underlying structure to represent the relationships between documents and keywords.

In this chapter, we present our conceptual work on graph data model. Section 5.2 contains the motivation for proposing the use of semantic groups of graph data model. In Section 5.3 we present the concept of semantic groups in a Graph Data Model. In Section 5.3.1 we show how semantic groups enhance sharing by allowing private views. We describe how semantic groups are affected by the graph operations in Section 5.3.3. We present an application of our graph data model in Section 5.4. We conclude with some discussions of the graph data model and semantic groups in Section 5.5.

5.2 Motivation

In this chapter, we present a graph data model, which focuses on defining views for sharing data while allowing context-sensitive updates on views. The fundamental basis of the approach is the concept of a semantic group [64] which has some similarities to the concept of views. A primary difference between the concept of semantic groups and the traditional concept of views is that views are defined on top of base relations while semantic groups are built as part of the data. For example, in this graph data model new data can be incorporated to an existing semantic group, without necessarily affecting some other semantic groups. The semantic groups provide more keyword-based semantic meaning than the traditional keywords' occurrences in

documents. This limits the scope of the query by the user in the presence of a very large number of documents.

The concept of views is important in database systems for controlling accesses and sharing of data. Traditional database systems are based on the concept of a schema, which defines the database using a model and facilitates querying of that database. Views are defined on top of base tables, and are implemented by query expressions. Any update to the base relations is reflected in the views. As a result, updates done within the context of a given view may affect other views. The focus of this chapter is to develop a new concept called "semantic group" for implementing views for sharing and querying in schema-less graph data model. Semantic groups allow updates to be made to private views. The proposed graph data model is suitable for modeling semi-structured data, such as documents. In the graph data model, nodes represent entities and links represent relationships between them. The notion of semantic groups has some similarities to that of views. The fundamental difference between semantic groups and traditional concept of views is that semantic groups are not query statements, but are part of the graph data itself. This allows us to make local updates easier. This approach is different from the object oriented approach, in which the relationships are defined between classes, and not necessarily between instances. These relationships should be standardized for all objects in a class, which may not be true in the case of semi-structured data. We introduce a two-layer graph structure: the basic layer that corresponds to the traditional graph data model, and a second layer that provides a mechanism for creating and updating views. The second layer implements the concept of *semantic groups*.

The graph data model and the semantic group concept can be used as a framework to represent clusters of documents. The idea is to build semantic groups based on output clusters and relationships to most important keywords. These relationships can then be reused in input ontology for subsequent clustering. This reuse of semantics is intended to incrementally improve the quality of clustering. Also, since clustering requires document processing and a structure to hold post-processed document data, the graph data model can also be used as an underlying structure to represent the relationships between documents and keywords and the corresponding weights. Update of this graph based structure is made easier by the concept of semantic groups.

5.2.1 Cluster Representation and Reuse

A semantic group would include an output cluster, encapsulated by a domain node, with links to the most relevant keywords. The order of relevance can be established by a statistical analysis of the documents in the clusters, based on keyword frequency within a document and the number of documents containing such keywords. Unlike dendrograms, this graph based representation of cluster keywords allows for overlapping categories.

For example, in Figure 5.1, the semantic group s_1 includes cluster c_1 and its most important keywords k_1, k_2, k_3, and semantic group s_2 includes c_2 and k_2, k_3, k_4. The links between keyword nodes can be based on the order of importance of keywords with respect to the corresponding clusters. The most relevant keywords obtained from the output cluster representation can be used to build new categories, which can define an input ontology for subsequent clusterings.

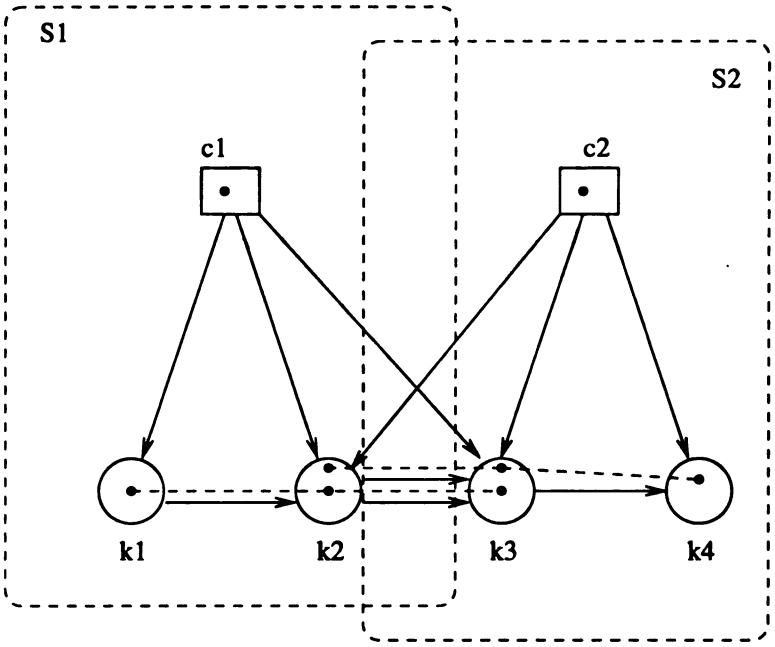


Figure 5.1: Cluster representation

The semantic groups of the graph data model, used for the management of ontologies, can handle the complexity of relationships between keywords, and the overlapping nature of such a graph. Multiple semantic groups, which are related to each other, can be merged to form a more general category. Querying capabilities of semantic group can be used to select and separate specific categories of an ontology for analysis and/or update.

5.2.2 Update of Private Views

Data sharing is important in database systems. Different parts of a large graph structure should be accessed and updated in different contexts. In the traditional approach, sharing is done through views. A traditional view is a dynamic subset of the database whose entities satisfy a specific logical condition. It is defined on top of

base relations. In our approach, we use the concept of semantic groups which are not defined on top of base tables. Here, all the data are created and updated in the context of semantic groups. Since a semantic group is not a full replication of the subgraph data, update is done directly to one's semantic group without having an unwanted side effect on other semantic groups. This is different from the materialized views approach [25], where physical copies are used to improve performance. Our approach is also different from the object oriented approach, where relationships are between classes, and not necessarily between instances.

5.3 Semantic Groups in Graph Data Model

The basic structure of a Graph Data Model, as presented in Chapter 2, is a graph where nodes correspond to entities, and labeled links represent the relationships between these entities. In addition to the concept of domain also presented in Chapter 2, the semantic group concept provides a contextual semantics for the components inside a domain. A semantic group is a set of components within a domain with a specific context. Thus, the same component in a domain can have different meaning in different contexts.

In the example of Figure 5.2, the "k_2" node can be linked to "d_1" node in the semantic group of "d_1", and be linked to "d_3" node in the semantic group of "d_3". Note that dashed links represent the semantic groups, whereas the solid arrows represent the user defined links. In this example, there are two semantic groups. The semantic group S_1 contains the nodes "d_1" and "d_2", and the links between them. And the

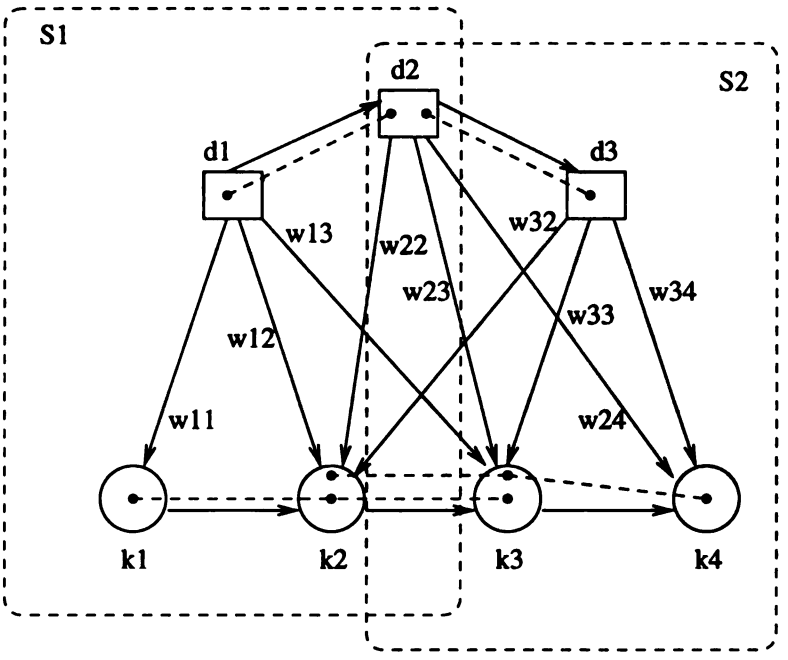


Figure 5.2: Example of semantic group

semantic group S_2 contains the nodes "d_2", "d_3", and the links between them.

To capture the concept of semantic groups we extend our definition of graph-based database in Chapter 2 as follows:

$$DB =< D, S, N, NT, NM, L, LT, LM >,$$

where the new element S is the set of all semantic groups. A domain can now be defined as $d = \{c_i\} \cup \{s_j\}$, where c_i is a component, and s_j is a semantic group within the domain. The other components of the graph-based database have similar meaning as in the previous definition.

5.3.1 Semantic Groups and Views

One important feature of semantic groups is the querying of graph structures. The *Select()* command is used to query a database in GDM.

Select(*inDomain*, *component*, *linkType*, *direction*, *depth*, *logicalOperator*, *outDomain*)

The main idea of *Select()* is to follow links, with a specified type and direction, within a semantic group, starting from a given component. The depth of the links followed can be controlled by the parameter "depth". Logical operators can be used for link specification. The link type, direction, depth and logical operator are optional. If a *Select()* command is applied to a domain with a unique semantic group and these optional parameters are not specified, then the output domain will contain the same elements as the input domain.

Select() command acts differently when semantic groups interact. Consider *Select()* with no optional parameters specified, and the input domain consists of several semantic groups. Unlike the case with a simple semantic group, the only components presented in the output domain are those which are in the same semantic groups as the starting component. That is how *Select()* command uses the notion of independent relations within the same component.

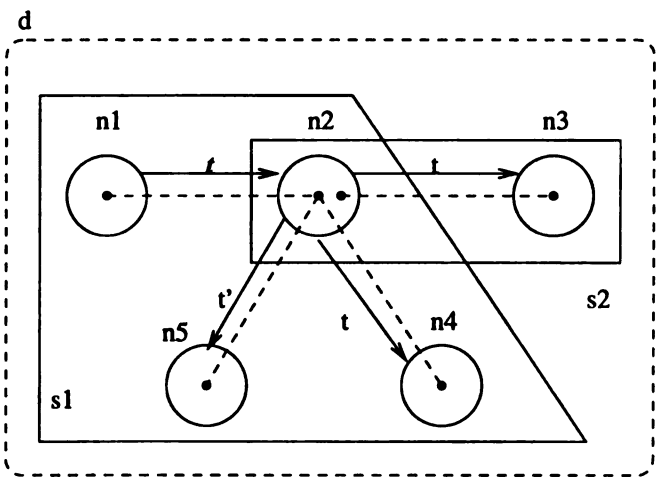


Figure 5.3: Querying based on semantic groups

The example in Figure 5.3 illustrates the use of *Select()* command in a domain with multiple semantic groups. The command $Select(d, n_1, t, to, null, null, d_1)$ will return the domain d_1 with nodes n_1, n_2 and n_4, and the links connecting them. Note that n_3 is not returned by the query because it is not in the same semantic group as the starting node n_1. On the other hand the node n_5 was not returned by the query because it is connected to n_2 with a link of type t' and not t.

5.3.2 Separation of Semantic Groups and Private Update

The strength of this data model consists of the creation and querying of semantic groups that contain shared components while allowing update of private data. This section shows how one can separate a semantic group so that private update can be done.

The *Separate(inDomain, outDomain)* command allows users to create independent semantic groups. It copies semantic groups from an input domain into an output domain specified by the user. These semantic groups will contain the same elements as the corresponding semantic groups in the input domain.

Figure 5.4 illustrates the *separate*() command. We start with domain d_1 of figure 5.4(a). After the operation $Separate(d_1, d_2)$ is applied on this domain d_1, the output domain d_2 with its semantic group s_2 becomes independent from the existing structure. Thus, this semantic group can be updated, i.e new nodes and links can be added to semantic group s_2, without affecting the domain d_1.

This flexibility in updating private views is especially beneficial when nodes consist of complex objects because components are not duplicated, but only subnodes

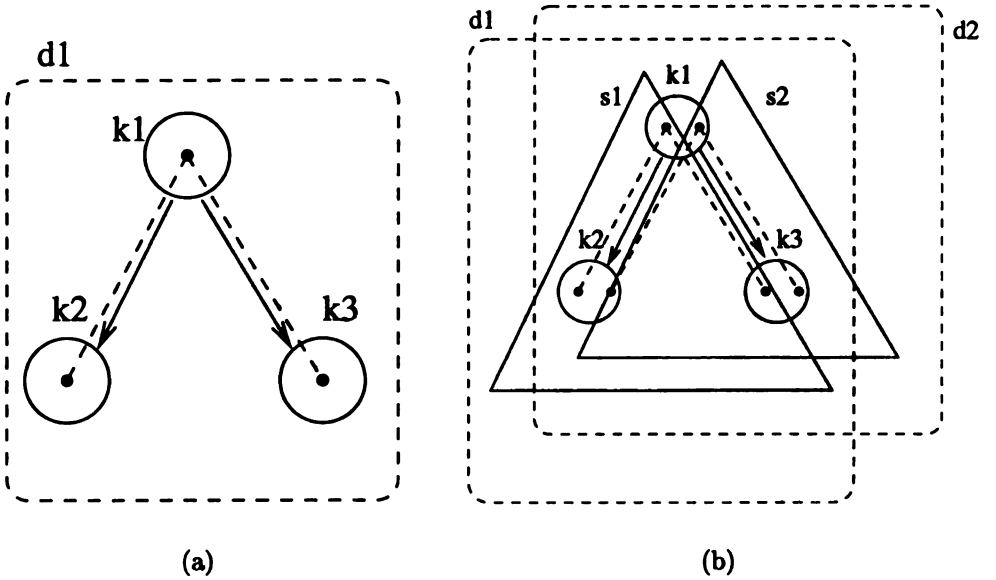


Figure 5.4: Separating semantic groups

corresponding to these nodes are created. The subnodes are used to implement the semantic groups. Nodes and links are associated to semantic groups through subnodes. Subnodes are hidden from users.

5.3.3 Effect of GDM Operations on Semantic Groups

In Section 5.3.2 we saw how one can separate semantic groups to be able to make updates to private views. This separation is possible because while components are created in a graph, the system creates the corresponding semantic groups. In this section we show how semantic groups are also captured through other operations of GDM, such as "create node" and "create link". A complete list of graph data model commands is included in Appendix B.

Semantic Groups and Newly Created Nodes

Let's consider a graph database $DB =< D, S, N, NT, NM, L, LT, LM >$.

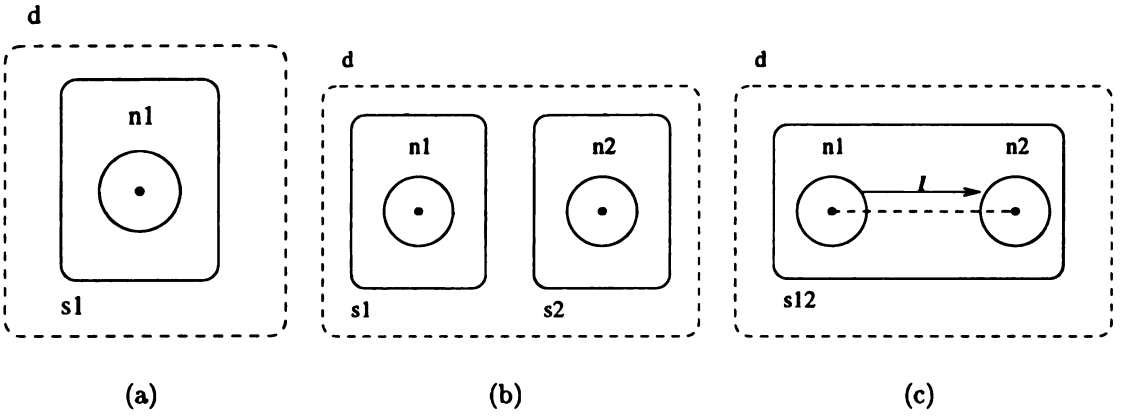
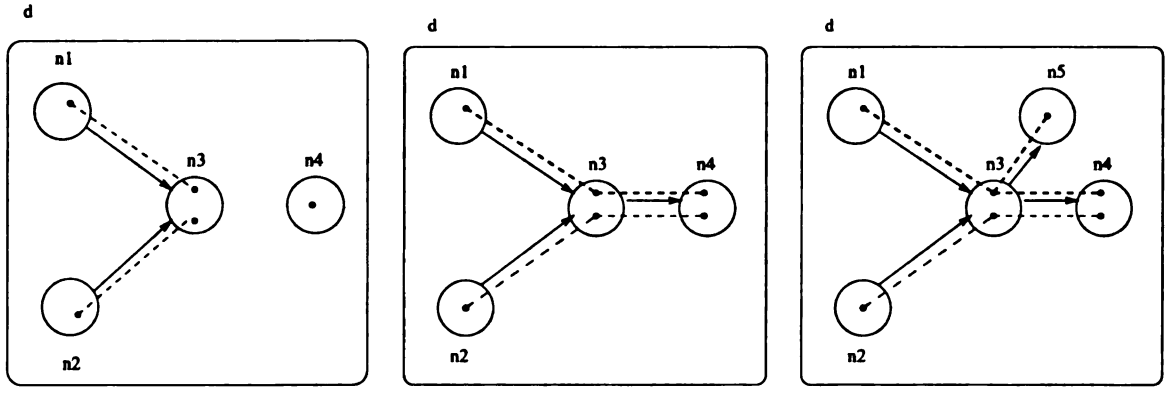


Figure 5.5: Semantic groups and basic graph operations

First we create a domain d, and a node n_1 within this domain. The system associates a semantic group $s_1 = \{n_1\}$ when the node n_1 is created (see Figure 5.5(a)). Initially, nodes are created with no link and belong to separate semantic groups. For example, in Figure 5.5(b) there are two semantic groups, $s_1 = \{n_1\}$ and $s_2 = \{n_2\}$. Nodes belong to the same semantic group when they are connected by links. For example, in Figure 5.5(c), after connecting the two nodes n_1 and n_2, the newly created semantic group is $s_{12} = \{n_1, n_2, l\}$. A more formal definition of linking two components is given in the next subsection.

Effect of Linking Components in Multiple Semantic Groups

The linking of two components that are members of multiple semantic groups is a more sophisticated operation than connecting components that belong to a single semantic group.



(a) (b) (c)

Figure 5.6: Connected nodes

Consider the example of Figure 5.6(a). The node n_3 is connected to n_1 and n_2 in respectively two different semantic groups. Now if we want to connect n_3 to n_4, we should be able to preserve the semantic groups associating n_4 to either n_1 or n_2, as shown in Figure 5.6(b). We may decide later that a new node n_5 linked to n_3 be associated to the semantic group $\{n_1, n_3, n_4\}$ and not $\{n_2, n_3, n_4\}$, as shown in Figure 5.6(c). Since a node can be shared in different contexts, all combinations of contexts should be preserved. A private update to this node can subsequently be done by separating the specific semantic group.

A union of semantic groups associated with the nodes being connected defines the semantic groups of the output domain. Every semantic group associated with the first linked node must be joined with every other semantic group associated with the other node.

More formally, consider a link l of type t created between nodes n_1 and n_2, respectively in domains d_1 and d_2. Moreover, suppose that n_1 belongs to the semantic

groups $s_{11}, s_{12}, ...$ and n_2 to $s_{21}, s_{22}, ...$ Then, the newly created link l is added to the set of links, i.e. $L \leftarrow L \cup \{l\}$, and the type t is added to the set LT of link types, $(LT \leftarrow LT \cup \{t\})$. The output domain is $d = d1 \cup d2 \cup \{l\}$. The set S of semantic groups in d is determined as follows:

Let $S_1 = \{s_{11}, s_{12}, ...\}$, $S_2 = \{s_{21}, s_{22}, ...\}$, and S' be the Cartesian product of S_1 and S_2, i.e., $S' = \{(s_{11}, s_{21}), (s_{11}, s_{22}), ...\}$. Then S consists of the union of $\{l\}$ with the sets in each pair of S', i.e., $S = \{s_{11} \cup s_{21} \cup \{l\}, s_{11} \cup s_{22} \cup \{l\}, ...\}$.

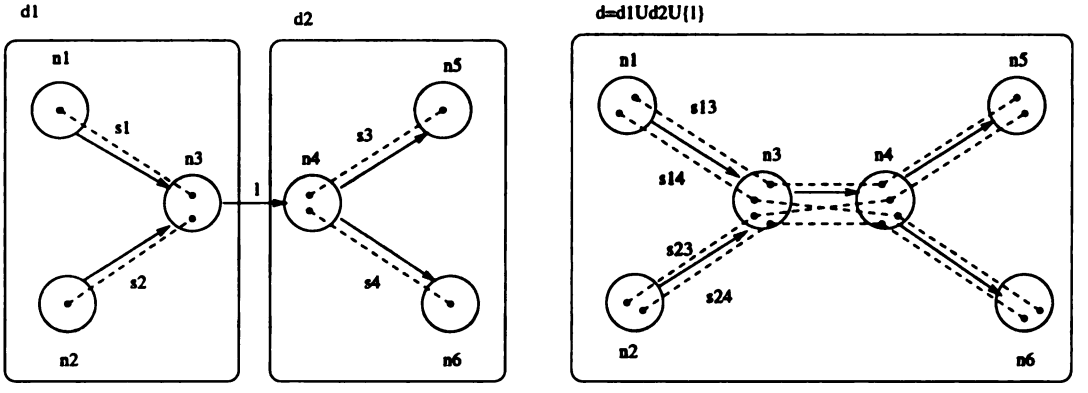


Figure 5.7: Linking nodes in interacting semantic groups

Figure 5.7 illustrates how two semantic groups $s_1 = \{n_1, n_3, l_{13}\}$ and $s_2 = \{n_2, n_3, l_{23}\}$ are combined with $s_3 = \{n_4, n_5, l_{45}\}$ and $s_4 = \{n_4, n_6, l_{46}\}$, when linking the two nodes n_3 and n_4 with link $l34$. This creates the four semantic groups: $s_{13} = s_1 \cup s_3 \cup \{l_{34}\} = \{n_1, n_3, l_{13}, n_4, n_5, l_{45}, l_{34}\}$, $s_{14} = s_1 \cup s_4 \cup \{l_{34}\} = \{n_1, n_3, l_{13}, n_4, n_6, l_{46}, l_{34}\}$, $s_{23} = s_2 \cup s_3 \cup \{l_{34}\} = \{n_2, n_3, l_{23}, n_4, n_5, l_{45}, l_{34}\}$ and $s_{24} = s_2 \cup s_4 \cup \{l_{34}\} = \{n_2, n_3, l_{23}, n_4, n_6, l_{46}, l_{34}\}$.

5.4 Semantic Groups and Documents

Typical Information Retrieval keyword searches return a large number of documents. Browsing through the whole set of retrieved documents may be a tedious task. Even though most Information Retrieval systems provide a way to tune the search, the protocol varies with each system. Some present users with categories of documents, so that searches can be performed within a subset of documents. However, these categories are static; and may not be up-to-date all the time. Querying document data is a difficult task because this data does not necessarily have a well defined schema that characterizes its structure. Therefore, it does not lend itself naturally to standard data models, such as relational model. A graph data model is more appropriate for such data. We propose the use of a graph-based system that provides a general framework for modeling documents. This system specifically uses the semantic group concept to provide a view-based framework for modeling documents. It adds more semantics to documents. Links represent semantic relationships between keywords or between documents and keywords. For instance, two or more documents may have information about the same subject of interest. Thus, they can be related in a semantic group. The weight of a keyword with respect to a given document is context-sensitive, in that it depends on the semantic group to which the relationship between a keyword and a document belong to. A graph-based system allows documents and keywords to be shared among several categories, or semantic groups.

Such a system may also allow for a search that returns a set of categories (concepts). These concepts may be different from topics in that they may not necessarily

be meaningful to the user. They are composed of a set of keywords with varying weights. These weights are context sensitive, in that the same keyword may carry different weights in the presence of two semantic groups.

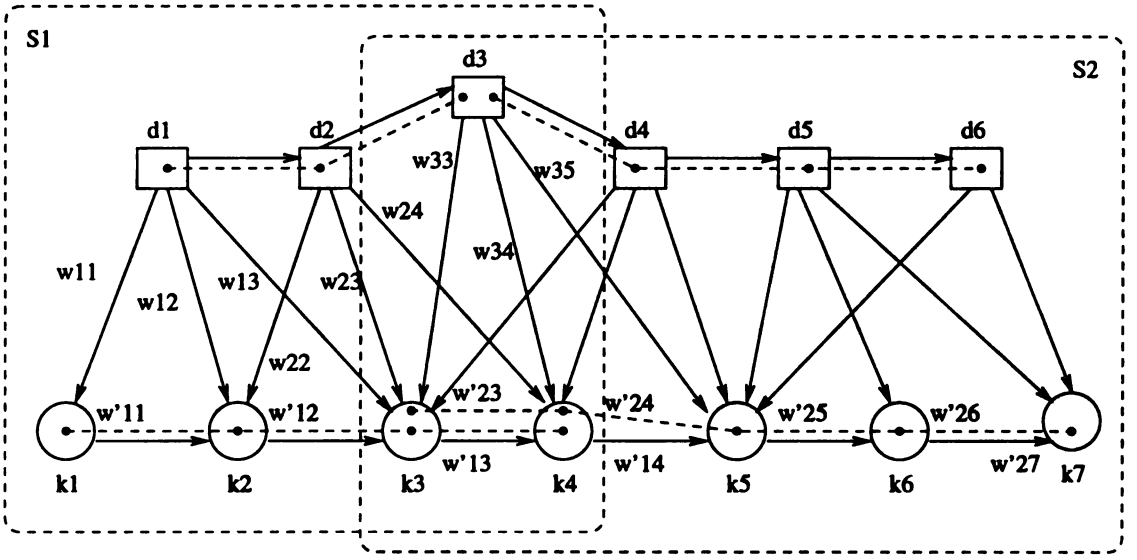


Figure 5.8: Context sensitivity using semantic groups

To illustrate this, let's consider Figure 5.8, where there are two semantic groups S_1 and S_2 defined as follows:

$S_1 = \{d_1, d_2, d_3, (d_1, d_2), (d_2, d_3), k_1, k_2, k_3, k_4, (k_1, k_2), (k_2, k_3), (k_3, k_4)\}$, and

$S_2 = \{d_3, d_4, d_5, d_6, (d_3, d_4), (d_4, d_5), (d_5, d_6), k_3, k_4, k_5, k_6, k_7, (k_3, k_4), (k_4, k_5), (k_5, k_6), (k_6, k_7)\}$,

where d_i refers to the ith document, k_i to the ith keyword, (d_i, d_j) to the link between d_i and d_j, and (k_i, k_j) to the link between k_i and k_j.

In Figure 5.8 dashed links represent the semantic groups, whereas the solid arrows represent the user defined links. Each link has an associated weight. For example, w_{11} refers to the weight of keyword k_1 in document d_1, and w'_{11} refers to the weight of keyword k_1 with respect to semantic group S_1. Note that d_3 is shared among the

two semantic groups S_1 and S_2. k_3 and k_4 are also shared among the two semantic groups. The weight of k_3 in S_1, w'_{13}, is carried in the subnode of k_3 in a sublink (k_2, k_3), and therefore is unseen in S_2. Whereas the weight of k_3 in S_2 is carried in the other subnode of k_3 in sublink (k_3, k_4), and therefore is unseen in S_1. The update of weights of keywords with respect to a category is done in the context of a specific semantic group. Thus, changes relative to a semantic group, such as the weight of k_3 with respect to group S_1, do not affect how k_3 is important to S_2. This has the advantage of having more stable categories. This is also important when a new document is inserted into the system. If keywords in this document are only relevant to one semantic group for example, then we do not have to worry about computing the weights of keywords that are not part of this semantic group. This makes the process more dynamic.

5.5 Summary

In this chapter, we presented the concept of semantic groups in a Graph Data Model, which provides a flexible mechanism for defining views while allowing update and incorporation of data to views. We proposed a mechanism by which semantic groups can be used in connection with document categorizing application. Since the weight of keywords depends on which category it is associated with, semantic groups can be used to capture this context-sensitivity. We showed that the semantic group concept can be used in situations where there is context sensitive information (in the above example, keyword weights within a given category), to make update more dynamic.

Chapter 6

Summary and Future Work

6.1 Summary

In this chapter we summarize the results achieved in this research work. We have defined an ontology based similarity measure (OBSM) that improves the quality of document clustering. Unlike traditional keyword based cosine vector similarity measure (CVSM), our measure takes into account an input ontology of keywords. The weight of a keyword is no longer only its frequency in a document, but is a function of its frequency and a context based coefficient. We evaluated the performance of OBSM first using a set of documents having pre-defined clusters, then using actual web-pages. We generated clustered data, where the clustered structure is measured by overlap in terms of common keywords. Then, we computed the dissimilarity between these clusters in terms of both CVSM and OBSM. Unlike for CVSM similarity, the ratio of intra-cluster over inter-cluster OBSM similarity is a non decreasing function of the intra-category overlap. This means that for two data sets, one exhibiting more

compact clusters than the other, OBSM will determine that the first data set has a better clustered structure. OBSM gave a slightly larger ratio of intra over inter-cluster similarity than CVSM, which means better cluster isolation.

We gathered all documents from the Yahoo medication index, and mixed in all documents that we got from the Detrol web site. This resulted in a database of preprocessed documents of dimension 9,397. We generated clustering dendrograms using complete link clustering for both CVSM and OBSM with varying weights. The results show that unlike CVSM, with proper choice of the weight of the keywords in the categories all Yahoo Detrol related web pages and Detrol.com documents cluster together when using OBSM.

We developed a system to generate synthetic data using spherical coordinates in high dimensional space to allow for control of cluster compactness and separation. This is unlike generating data in the Cartesian coordinate system, where controlling parameters for a desired distribution in terms of angles, is not easily achievable by Cartesian parameters. We used normal distribution with various standard deviations to scatter documents within clusters around centroid polar coordinates. We also used angle based overlap to specify cluster isolation. This allows the control of inter cluster overlap.

We proposed the use of the concept of semantic groups of graph data model and presented a methodology to integrate documents, categories and the corresponding keyword weights, by mapping categories to semantic groups. Semantic group concept can be used in situations where there is context sensitive information to make update more dynamic.

6.2 Future Work

As the need for querying data over the Web is growing, users are becoming more demanding, and the volume of data on the web increases. Any improvement of similarity measure between documents will contribute to improve search engine performance. In this work we assumed that an ontology existed for the similarity measure to use. In order for ontology based similarity measure to gain wide use, a method needs to be developed to build and/or identify this ontology, preferably with minimum human intervention. As seen in Chapter 3, our tests on a real web data set showed that the choice of category weights does affect the quality of clusters produced by ontology based similarity measure. This effect warrants further analysis for the purpose of understanding how to choose the best weights. For example, our method of synthetic data generation in polar coordinate system can be applied to investigate the relation between data characteristics and the performance of category weights. The ultimate test for the concept of semantic groups in graph data model is to be implemented in an integrated information retrieval system that uses our ontology-based similarity measure as part of its indexing scheme.

APPENDICES

Appendix A

Some Web Pages from Data Set

In this appendix, we include snapshots of some of the web pages that were used in the real data set. Figure A.1 shows Yahoo's "Drugs and Medications" category[1] page. Figure A.2 shows a part of the "D" medication index[2] of Yahoo. Figure A.3 shows Detrol description[3] page.

[1] http://dir.yahoo.com/Health/Pharmacy/Drugs_and_Medications/
[2] http://health.yahoo.com/health/pdr_drugs/d.html
[3] http://health.yahoo.com/health/pdr_drugs/0908/0.html

YAHOO! powered by COMPAQ Help - Check Email

FREE Profile: Height 5 ft 7 in Weight Go!

Click Here to Lose 10 lbs - START NOW!

Yahoo! Directory
Drugs and Medications

Search Advanced Search Help
all of Yahoo! just this category

Home > Health > Pharmacy > **Drugs and Medications**

Inside Yahoo!

- **Yahoo! Health: Drugs and Medications** - definitions, alternative names, and related resources.

Categories

- **Booksellers@**
- **Culture** *(19)*
- **Drug Interactions** *(14)* NEW!
- **Drug Policy** *(59)*
- **Organizations** *(7)*
- **Psychopharmacology@**
- **Publications** *(5)*
- **Research** *(1)*
- **Resources** *(33)*
- **Specific Drugs and Medications** *(825)* NEW!
- **Substance Abuse@**
- **Types** *(258)* NEW!
- **FAQs** *(1)*
- **Usenet** *(10)*

More Yahoo!

News: Antibiotics and Microbiology

News: Drug Trade

News: Pharmaceutical Industry

Email this Category to a Friend

Get Our NEW BOOKLET of the 101 MOST USEFUL WEBSITES FREE - CLICK HERE

Figure A.1: Yahoo's "Drugs and Medications" category page

. Delavirdine (Oral)
. Delaxin®
. Delsym Hold®
. Delsym®
. Deltalin®
. Deltasone®
. Demadex®
. Demerol® (Injection)
. Demerol® (Oral)
. Demi-Regroton®
. Demulen® 1/50
. Denavir®
. Denileukin Diftitox (Injection)
. Depakene®
. Depakote®
. Depen®
. Depo-Provera®
. Depo-Testosterone®
. Depocyt(Tm)
. Deponit®
. Depotest®
. Dermabet®
. Dermacomb®
. Dermtex Hc®
. Deserpidine/Thiazide Diuretics (Oral)
. Desferal®
. Desiccated Thyroid (Oral)
. Desipramine (Oral)
. Desmopressin (Injection)
. Desmopressin (Nasal)
. Desogen®
. Desoximetasone (Topical)
. Desoxyn®
. Desquam-E(Tm)
. Desquam-X®
. Desyrel®
. Detrol®
. Dexacidin®
. Dexair®
. Dexamethasone (Injection)
. Dexamethasone (Ophthalmic)
. Disipal®
. Disonate®
. Disopyramide (Oral)
. Disulfiram (Oral)
. Ditropan®
. Diucardin®
. Diulo®
. Diupres®
. Diurese-R®
. Diuril®
. Diutensin-R®
. Dobutamine (Injection)
. Dobutrex®
. Docetaxel (Injection)
. Docusate
. Dofetilide (Oral)
. Dolobid®.
. Dolophine®
. Dolorac®
. Donatussin®
. Donepezil (Oral)
. Donnagel®
. Donnatal Exentabs®
. Donnatal®
. Donnazyme®
. Doral®
. Dornase Alfa (Inhalation)
. Doryx®
. Dorzolamide (Ophthalmic)
. Doss
. Dovonex®
. Doxazosin (Oral)
. Doxepin (Oral)
. Doxercalciferol (Oral)
. Doxil®
. Doxinate D-S-S®
. Doxorubicin (Injection)
. Doxorubicin Liposomal (Injection)
. Doxycycline (Oral)
. Dramamine®
. Drisdol®

Figure A.2: Part of the "D" medication index

YAHOO! Health

Yahoo! - Help

PDR.net | PDR FAMILY GUIDES

Free
Fetal Development Updates

Yahoo! Health

Health Hc

Home > Medication or Drug

Search

Medication or Drug

Search alphabetically

* Diseases & Conditions
* Find Clinical Trials

Yahoo! Categories

* Drugs and Medications
* Pharmaceutical Companies
* Pharmacies
* Substance Abuse

Yahoo! Community

* Events: Drugs and Medications
* Groups: Drugs and Medications
* Message Boards: Drugs and Medications

Yahoo! Resources

* Find a Doctor
* Health Assessment Test
* Yellow Pages: Pharmacies

Tolterodine (Oral)

Overview | Precautions

Description

Helps patients with an overactive bladder control their urine. Also helps control the feeling of having to urinate right away or having to urinate too often.

Brand Name(s)

Detrol®

When you should not use this medicine

You should not use this medicine if you had an allergic reaction to tolterodine. You should not use this medicine if you have problems passing urine, gastric retention (problems with food emptying from your stomach), or uncontrolled glaucoma.

How to use and store this medicine

Tablets:

- Your doctor will tell you how much to take and how often.
- You may take this medicine with or without food.
- Store the tablets at room temperature, away from heat, moisture, and direct light.
- Keep all medicine out of the reach of children.

If you miss a dose:

- Take the missed dose as soon as possible, unless it is almost time for your next dose.
- Skip the missed dose if it is almost time for your next regular dose.
- You should not use two doses at the same time.

Figure A.3: Detrol description page

Appendix B

Commands of Graph Data Model

- *CreateNode(inputDomain, type, name, outputDomain)*

  This command creates a node with the given (unique) *name* and *type* in *outputDomain*. A new semantic group is created and associated with the node. All data from *inputDomain* is copied into *outputDomain*.

- *CreateLink(inputDomain1, node1, inputDomain2, node2, type, direction, outputDomain)*

  This command creates a link, of the given *type*, between the nodes *node1* and *node2*, in *outputDomain*. A new semantic group will be created in *outputDomain*, and contains the Cartesian product of all semantic groups (with the specified nodes in the input domains) and the newly created link. The content of both input domains is copied into the output domain. *Direction* could be *from* or *to*. If the link is semantically symmetric, option *none* can be used.

- *DeleteNode(inputDomain, name, outputDomain)*

  This command copies all data from *inputDomain* into *outputDomain*, then deletes the node from all semantic groups in *outputDomain*. Dangling links may be created.

- *DeleteLink(inputDomain, node, node, type, direction, outputDomain)*

  This command copies all data from *inputDomain* into *outputDomain*, then deletes the link from all applicable semantic groups in *outputDomain*. Isolated nodes may be created.

- *Separate(inputDomain, outputDomain)*

  This command creates copies of all semantic groups from *inputDomain* to *outputDomain*. It does not copy the content of the input domains. The input and output domains for this command must be different.

- *SeparateComponent(inputDomain, component, outputDomain)*

  This command creates a new semantic group for the specified component in *outputDomain*. It is similar to the *separate* command. However, since this command was very useful for users of the system, it was included into the command set.

- *Select(inputDomain, component, linkType, direction, depth, logicalOperator, outputDomain)*

  This command selects parts (or all) of the semantic groups that contain the specified elements from the input domains, and puts them into *outputDomain*.

This command is essentially a generalized transitive closure with respect to the *select* parameters. Starting from the initial component, all components connected by the links of the specified types, directions, depth (i.e. the number of links from the initial node), and corresponding semantic groups are selected into it outputDomain. For the links' specification, logical operators ("*and*", "*or*", "*not*") can be used. Special reserved keywords indicate two boundary situations. When no link type should be considered, the keyword "*none*" is used; "*all*" indicates that all link types will be followed. Unlike *separate()* command, *select()* provides filtering only, i.e. components that satisfy some conditions are put into the output domain, whereas *separate* creates new semantic groups, different from initial groups, although they contain the same components.

- *Union({inputDomains}, outputDomain)*

  This command takes the union of all semantic groups in all input domains and places their union into *outputDomain*. If a component exists in multiple input domains, it will be presented in *outputDomain* only once. If a component belongs to multiple semantic groups they will be presented in *outputDomain*.

- *Intersection({inputDomains}, outputDomain)*

  The command takes the intersection of the sets of components in the input domains, retrieves all semantic groups that correspond to the components in the input domains, and places them in *outputDomain*. All components that exist in all input domains will be presented in *outputDomain*, even if they belong to different semantic groups.

- *CreateDomain(inputDomain, domainName, outputDomain)*

  This command creates an empty domain "*domainName*" in *outputDomain*. The output domain will also contain all components in *inputDomain*.

- *DeleteDomain(inputDomain, domainName, outputDomain)*

  This command deletes the domain "*domainName*" and all components that only belong to this domain. The remaining components are copied to *outputDomain*.

BIBLIOGRAPHY

[1] S. Abiteboul, "Querying semi-structured data," in *Proceedings of the International Conference on Database Theory (ICDT)*, pp. 1–18, Springer-Verlag, Jan. 1997.

[2] S. Abiteboul, S. Cluet, V. Christophides, T. Milo, G. Moerkotte, and J. Simeon, "Querying documents in object databases," *International Journal on Digital Libraries*, vol. 1, Apr. 1997.

[3] C. C. Aggarwal, C. Procopiuc, J. Wolf, P. S. Yu, and J. Park, "Fast algorithms for projected clustering," *SIGMOD'99*, pp. 61–72, 1999.

[4] C. C. Aggarwal and P. S. Yu, "Finding generalized projected clusters in high dimensional spaces," in *Proceedings of the International ACM SIGMOD Conference on Management of Data*, (Dallas, TX), pp. 70–81, ACM Press, May 2000.

[5] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan, "Automatic subspace clustering of high dimensional data for data mining applications," *SIGMOD'98*, pp. 94–105, 1998.

[6] R. Armstrong, D. Freitag, T. Joachims, and T. Mitchell, "WebWatcher: A learning apprentice for the World Wide Web," *Proceedings of the 1995 AAAI Spring Symposium on Information Gathering from Heterogeneous, Distributed Environments*, Mar. 1995.

[7] P. Atzeni, G. Mecca, and P. Merialdo, "To weave the Web," in *Proceedings of the International Conference on Very Large Data Bases*, (Athens, Greece), 1997.

[8] R. Baeza-Yates and B. Ribeiro-Neto, *Modern Information Retrieval.* Addison Wesley, 1999.

[9] K. Bharat and G. A. Mihaila, "When experts agree: Using non-affiliated experts to rank popular topics," in *The International World Wide Web Conference*, pp. 597–602, May 2001.

[10] R. Botafogo, "Cluster analysis for Hypertext systems," in *the Proceedings of ACM SIGIR'93*, 1993.

[11] P. Buneman, "A query language and optimization techniques for unstructured data," in *SIGMOD'96*, (Montreal, Canada), pp. 505–516, 1996.

[12] P. Buneman, "Semistructured data," in *ACM PODS'97*, (Tucson, Arizona USA), pp. 117–121, 1997.

[13] T. Catarci, S. Chang, M. Costabile, S. Levialdi, and G. Santucci, "A graph-based framework for multiparadigmatic visual access to databases," *IEEE Transactions on Knowledge and Data Engineering*, vol. 8, no. 3, pp. 445–475, 1996.

[14] P. Chen, "The Entity-Relationship Model - Toward a unified view of data," *ACM Transactions on Database Systems*, vol. 1, no. 1, pp. 9–36, 1976.

[15] M. P. Consens and A. O. Mendelzon, "The G+/GraphLog visual query system," in *Proceedings of the International ACM SIGMOD Conference on Management of Data*, (Atlantic City, NJ), p. 388, May 1990.

[16] M. P. Consens and A. O. Mendelzon, "Hy^+: A hygraph-based query and visualization system," *SIGMOD Record*, vol. 22, pp. 511–516, June 1993.

[17] W. H. E. Day, "Efficient algorithms for agglomerative hierarchical clustering methods," *Journal of Classification*, vol. 1, no. 1, pp. 7–24, 1984.

[18] D. Filo and J. Yang, "Yahoo!," 1995. http://www.yahoo.com.

[19] D. Fisher, "Iterative optimization and simplification of hierarchical clusterings," *Journal of Artificial Intelligence Research*, vol. 4, pp. 147–179, 1996.

[20] C. Fox, "Lexical analysis and stoplists," in *Information Retrieval Data Structures and Algorithms* (W. B. Frakes and R. Baeza-Yates, eds.), pp. 102–130, Englewood Cliffs, New Jersey: Prentice Hall, 1992.

[21] W. B. Frakes, "Stemming algorithms," in *Information Retrieval Data Structures and Algorithms* (W. B. Frakes and R. Baeza-Yates, eds.), pp. 131–160, Englewood Cliffs, New Jersey: Prentice Hall, 1992.

[22] W. B. Frakes and R. Baeza-Yates, *Information Retrieval Data Structures and Algorithms*. Englewood Cliffs, New Jersey: Prentice Hall, 1992.

[23] M. Franz and J. S. McCarley, "Unsupervised and supervised clustering for topic tracking," in *Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, (New Orelans, Louisiana), pp. 310–317, ACM Press, Sept. 2001.

[24] M. Gemis, J. Paredaens, I. Thyssens, and J. V. Bussche, "GOOD: A graph-oriented object database system," *SIGMOD Record*, vol. 22, pp. 505–510, June 1993.

[25] T. Griffin and L. Libkin, "Incremental maintenance of views with duplicates," in *Proceedings of the International ACM SIGMOD Conference on Management of Data*, vol. 24, (San Jose, CA USA), pp. 328–339, June 1995.

[26] S. Guha, R. Rastogi, and K. Shim, "CURE: An efficient clustering algorithm for large databases," *SIGMOD'98*, pp. 73–84, 1998.

[27] M. Gyssens, J. Paradaens, and D. V. Gucht, "A graph-oriented object model for database end-user interfaces," *SIGMOD Record*, vol. 19, no. 2, pp. 24–33, 1990.

[28] J. Hammer, H. Garcia-Molina, K. Ireland, Y. Papakonstantinou, J. Ullman, and J. Widom, "Information translation, mediation, and mosaic-based browsing in the TSIMMIS system," in *Proceedings of the International ACM SIGMOD Conference on Management of Data*, vol. 24, (San Jose, CA USA), p. 483, June 1995.

[29] M. A. Hearst, D. R. Karger, and J. O. Pedersen, "Scatter/Gather as a tool for the navigation of retrieval results," in *The Working Notes of the 1995 AAAI Fall Symposium on AI Applications in Knowledge Navigation and Retrieval*, 1995.

[30] M. A. Hearst and J. O. Pedersen, "Reexamining the cluster hypothesis: Scatter/Gather on retrieval results," in *Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, (Zurich, Switzerland), pp. 76–84, Aug. 1996.

[31] "Ht://Dig," 1995. http://www.htdig.org.

[32] M. Iwazume, K. Shirakami, K. Hatadani, H. Takeda, and T. Nishida, "IICA: An ontology-based internet navigation system," in *Working notes for AAAI-96 Workshop on Internet-Based Information Systems*, pp. 65–71, 1996.

[33] M. Iwazume, H. Takeda, and T. Nishida, "Ontology-based approach to information gathering and text categorization," in *Proceedings of International Symposium on Digital Libraries*, pp. 186–193, 1995.

[34] A. K. Jain and R. C. Dubes, *Algorithms for clustering data.* Prentice Hall, 1988.

[35] T. Joachims, T. Mitchell, D. Freitag, and R. Armstrong, "WebWatcher: Machine learning and Hypertext," *Fachgruppentreffen Maschinelles Lernen*, Aug. 1995.

[36] A. Johnson, A. Ramachandran, and S. Pramanik, "Efficacy: A data gathering system for the World Wide Web," Master's thesis, Computer Science and Engineering Department, Michigan State University, July 1998.

[37] S. C. Johnson, "Hierarchical clustering schemes," *Psychometrika*, vol. 32, pp. 241–254, 1967.

[38] N. Kiesel, A. Schuerr, and B. Westfechtel, "GRAS, a graph-oriented (software) engineering database system," *Information Systems*, vol. 20, no. 1, pp. 21–51, 1995.

[39] B. King, "Step-wise clustering procedures," *Journal of the American Statistical Association*, vol. 69, pp. 86–101, 1967.

[40] D. Konopnicki and O. Shmueli, "W3QS: A query system for the World Wide Web," in *Proceedings of the International Conference on Very Large Data Bases*, (Zurich), pp. 54–65, 1995.

[41] H. S. Kunii, *Graph Data Model and its Data Language*. Springer-Verlag, 1990.

[42] L. Lakshmanan, F. Sadri, and I. Subramanian, "A declarative language for querying and restructuring the Web," in *6th Intern. Workshop on Research Issues in Data Engineering: Interoperability of Nontraditional Database Systems*, 1996.

[43] D. Lawrie, W. B. Croft, and A. Rosenberg, "Finding topic words for hierarchical summarization," in *Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, (New Orelans, Louisiana), pp. 349–357, ACM Press, Sept. 2001.

[44] M. Levene and G. Loizou, "A graph-based data model and its ramifications," *IEEE Transactions on Knowledge and Data Engineering*, vol. 7, pp. 809–823, Oct. 1995.

[45] D. D. Lewis, "Evaluating text categorization," in *Proceedings of Speech and Natural Language Workshop*, pp. 312–318, Defense Advanced Research Projects Agency, Morgan Kaufmann, Feb. 1991.

[46] D. D. Lewis, "Feature selection and feature extraction for text categorization," in *Proceedings of Speech and Natural Language Workshop*, (San Mateo, CA), pp. 212–217, Defense Advanced Research Projects Agency, Morgan Kaufmann, 1992.

[47] D. D. Lewis and W. B. Croft, "Term clustering of syntactic phrases," in *Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 385–404, 1990.

[48] D. D. Lewis and W. A. Gale, "A sequential algorithm for training text classifiers," in *Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, (Dublin, Ireland), pp. 3–12, July 1994.

[49] D. Lucarella and A. Zanzi, "A visual retrieval environment for hypermedia information systems," *ACM Transactions on Information Systems*, vol. 14, pp. 3–29, Jan. 1996.

[50] "Lycos." http://lycos.cs.cmu.edu.

[51] J. D. Martin, "Clustering full text documents," in *Proccedings of IJCAI-95 Workshop on Data Engineering for Inductive Learning*, (Montreal, Canada), Aug. 1995.

[52] F. Menczer, G. Pant, P. Srinivasan, and M. E. Ruiz, "Evaluating topic-driven Web Crawlers," in *Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, (New Orelans, Louisiana), pp. 241–249, ACM Press, Sept. 2001.

[53] A. O. Mendelzon, G. A. Mihaila, and T. Milo, "Querying the World Wide Web," in *Proceedings of the First International Conference on Parallel and Distributed Information Systems*, pp. 80–91, Dec. 1996.

[54] G. A. Miller, "Introduction to WordNet: An on-line lexical database," Aug. 1993.

[55] G. A. Miller, "WordNet: A lexical database for english," *Communications of the ACM*, pp. 39–41, Nov. 1995.

[56] C. T. Mosier, "An experiment investigating the application of clustering procedures and similarity coefficients to the gt machine cell formation problem," *International Journal of Production Research*, vol. 27, no. 10, pp. 1811–1835, 1989.

[57] F. Murtagh, "A survey of recent advances in hierarchical clustering algorithms," *The Computer Journal*, vol. 26, no. 4, pp. 354–359, 1983.

[58] C. Ordonez and P. Cereghini, "SQLEM: Fast clustering in sql using the em algorithm," in *Proceedings of the International ACM SIGMOD Conference on Management of Data*, (Dallas, TX), pp. 559–570, ACM Press, May 2000.

[59] J. Paredaens, P. Peelman, and L. Tanca, "G-Log: A graph-based query language," *IEEE Transactions on Knowledge and Data Engineering*, vol. 7, p. 436, June 1995.

[60] P. Pirolli, P. Schank, M. Hearst, and C. Diehl, "Scatter/Gather browsing communicates the topic structure of a very large text collection," in *Human Factors in Computing Systems CHI '96*, pp. 213–220, Association for Computing Machinery, 1996.

[61] M. Porter, "An algorithm for suffix stripping," *Program*, vol. 14, no. 3, pp. 130–137, 1980.

[62] A. Poulovassilis and M. Levene, "A nested-graph model for the representation and manipulation of complex objects," *ACM Transactions on Information Systems*, vol. 12, pp. 35–68, Jan. 1994.

[63] S. Pramanik, S. Alexander, and J. Li, "Efficient searching algorithms for approximate nearest neighbor queries in high dimensions," in *IEEE Multimedia Int. Conf. on Multimedia Computing and Systems*, pp. 865–869, 1999.

[64] S. Pramanik, N. Assem, S. Choudhury, and S. Bhattacharya, "Graph theoretic modeling of semi-structured information system based on functional abstraction," in *Proceedings of the IASTED International Conference on Applied Modeling Systems IASTED-AMS'98*, (Honolulu, Hawaii - USA), pp. 518–522, Aug. 1998.

[65] S. Pramanik, J. Li, and J. Ruan, "Physics of high dimensional data sets in euclidean space and their applications," in *SSGRR-2000*, (L'Aquila, Italy), July 2000.

[66] C. Y. Quek, "Classification of World Wide Web documents," Master's thesis, School of Computer Science, Carnegie Mellon University, 1997.

[67] E. Rasmussen, "Clustering algorithms," in *Information Retrieval Data Structures and Algorithms* (W. B. Frakes and R. Baeza-Yates, eds.), pp. 419–442, Englewood Cliffs, New Jersey: Prentice Hall, 1992.

[68] S. E. Robertson and K. S. Jones, "Relevance weighting of search terms," *Journal of the American Society for Information Sciences*, vol. 27, no. 3, pp. 129–146, 1976.

[69] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information Processing & Management*, vol. 24, pp. 513–523, 1988.

[70] G. Salton, E. A. Fox, and H. Wu, "Extended boolean information retrieval," *Communications of the ACM*, vol. 26, no. 11, pp. 1022–1036, 1983.

[71] S. M. Shaffer and D. F. Rogers, "Similarity and distance measures for cellular manufacturing. part II. an extension and comparison," *International Journal of Production Research*, vol. 31, no. 6, pp. 1315–1326, 1993.

[72] L. G. Shapiro and G. C. Stockman, *Computer Vision*. Prentice Hall, 2001.

[73] P. C. Shields, *Linear Algebra*. Addison Wesley, 1966.

[74] R. R. Sokal and F. J. Rohl, "The comparison of dendrograms by objective methods," *Taxon*, vol. XI, pp. 33–39, Feb. 1962.

[75] A. J. Vakharia and U. Wemmerlov, "A comparative investigation of hierarchical clustering techniques and dissimilarity measures applied to the cell formation problem," *Journal of Operations Management*, vol. 13, pp. 117–138, Feb. 1995.

[76] P. Willett, "Recent trends in hierarchic document clustering: a critical review," *Information Processing & Management*, vol. 24, no. 5, 1988.

[77] M. Wulfekuhler and W. Punch, "Finding salient features for personal Web page categories," in *The International World Wide Web Conference*, (Santa Clara, CA), Apr. 1997.

[78] Y. Yang, "Sampling strategies and learning efficiency in text categorization," in *AAAI Spring Symposium on Machine Learning in Information Access*, (Stanford), Mar. 1996.

[79] B. Yumono and D. Lee, "WISE: A World Wide Web resource database system," *IEEE Transactions on Knowledge and Data Engineering*, pp. 548–554, Aug. 1996.

[80] O. Zamir and O. Etzioni, "Web document clustering: A feasibility demonstration," in *Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, (Melbourne, Australia), pp. 46–54, ACM Press, 1998.

[81] O. Zamir, O. Etzioni, O. Madani, and R. M. Karp, "Fast and intuitive clustering of Web documents," in *The International Conference on Knowledge Discovery and Data Mining*, 1997.

[82] T. Zhang, R. Ramakrishnan, and M. Livny, "BIRCH: An efficient data clustering method for very large databases," *SIGMOD'96*, pp. 103–114, 1996.