A STUDY ON DOCUMENT RELATION DISCOVERY USING FREQUENT ITEMSET MINING

A Thesis Presented

by

Kritsada Sriphaew

Doctor of Philosophy

Information Technology Program

Sirindhorn International Institute of Technology

Thammasat University

May 2007

A STUDY ON DOCUMENT RELATION DISCOVERY USING FREQUENT ITEMSET MINING

A Thesis Presented

By

Kritsada Sriphaew

Submitted to

Sirindhorn International Institute of Technology

Thammasat University

In partial fulfillment of the requirement for the degree of

DOCTOR OF PHILOSOPHY IN TECHNOLOGY

Approved as to style and content by the Thesis Committee:

Chair and Advisor

Assoc. Prof. Thanaruk Theeramunkong, Ph.D.

Co-Advisor

Assoc. Prof. Stanislav S. Makhanov, Ph.D.

Member

Assoc. Prof. Ekawit Nantajeewarawat, Ph.D.

Member

Asst. Prof. Junalux Chalidabhongse, Ph.D.

May 2007

Acknowledgment

People can not walk through the hardship alone, they need several supporters on the way to reach the destination. Including me, I cannot succeed if I did not get supports from these following hands.

Being a teacher is not that easy, but Dr.Thanaruk Theeramunkong shows me more than a great teacher. He has become an important person in my study life. Starting from giving me a scholarship to study, advising how to do research for me from the beginning, giving supports whenever I wanted, writing paper together until morning, showing his gentle mind and kindness that I have never seen from anyone, these have been the invaluable things which I had received all along my six years. Deeply grateful may not enough for my precious advisor.

I also spend my gratitude to the co-advisor and committee members, Dr. Stanislav S. Makhanov, Dr. Ekawit Nantajeewarawat and Dr. Junalux Chalidabhongse, who gave several good comments from their large professional and technical skills during the progress presentation all along the six years and the thesis defense. My gratitude also spends to the external examiner, Prof.Tu-Bao Ho, for his kindness in examining this thesis, including all editors and reviewers who used to revise my publications concerning to this work. Besides, Mr. Terrance Downey, a native English teacher, who helps to proofread my English writing is also thankful. Also many thanks to the two IT lecturers, Dr. Cholwich Nattee and Dr. Pakinee Suwannajan for their helps in the evaluation process of this work and useful suggestions in laboratory's meeting.

For the Okumura Laboratory in Japan, my deep thank is spent to Okumura-sensei who gave me a great opportunity to visit his laboratory as a visiting researcher, and all laboratory's members there who looked after me very well. I had learned several aspects during my five-months stay in Japan. I am so proud to be in a part of their laboratory although it is just a short period. Thank you for the experiences and friendships which are unforgettable.

Furthermore, I also thank to Thai Computational Linguistic (TCL) Laboratory and National Electronics and Computer Technology Center (NECTEC) in collaborating with my laboratory for doing several research activities, such as organizing the conferences, organizing the software contests and supporting me for presenting research works in the international conferences. I have gained a lot of invaluable experiences through those activities.

Besides, the seniors of graduated student in IT school, i.e., P'Tee, P'Jum, P'Mai, P'Jack, P'Sanon, P'Kai, P'A+, P'Lee, P'Tum, P'Winnie and Nok, are also helpful in giving many kind suggestions and accept me in their levels. Another mentions are referred to the seniors of graduated student in IE school, i.e., P'Aw, P'Pae and P'Yong, including all juniors of IT, IE, MT and TC schools for being friends to hang out during my study. Thanks for their support and humor.

To my deep friends whom I found during study, P'Som (Angel), P'Kaew (Gaussian) and P'Tum (Hippo), we had shared the unforgettable memory together. Staying late until

morning, suffering from what we had encountered, giving supports without uttering any word, traveling together when we had several deadlined works, etc. I am indeed appreciate that we had each other whether the happiness time or unhappiness moment. We know how strong of our friendship is, and only thank word is not enough for them. Other friends which I did not mention here are also thankful. Although they did not know much about what I did during these six years, they gave many good supports whenever I needed.

Many warm hugs to my family who always understand what I had been doing. They are far from knowing what the research is, but they keep staying beside me to let my dream comes true. It would be troublesome if they request me to superintend the family, such as earning the money, doing social activity or traveling with them in the holidays, but those are not necessary for them except my happiness. Thank you so much for non-making strain on me but supporting me instead. With honestly, I love them.

To myself, thank you for the research imagination which still exists although it became less when I had been getting down. I have learned that the inner feeling is a main factor for doing everything. "*If your inner still feels to fight, you will also fight whatever the worst situation you are encountered*." And, I did fight!

Last, but not least, a great gratitude is delivered to my main financial supporter, the Royal Golden Jubilee (RGJ) Ph.D. Program of Thailand Research Fund (TRF). This program gives me, the new generations, an opportunity to have higher education. With deep thank to this program, I promise to use all of my knowledge to help improving Thailand and contributing positive benefits to the cosmopolite as much as possible until the end of my days.

Abstract

Scientific publications available in the digital libraries are potentially the world's largest knowledge source but there have been very few attempts to take advantage of this kind of document. One traditional knowledge which is useful for retrieving desired information, understanding the nature of document contents and revealing hidden information between a set of documents, is the relations among such a collection of documents. Although relations among technical documents are distinctively useful, there is no trustworthy automatic approach to evaluate the quality of discovered relations. Extended from a relationship between a document pair, the document relation can involve more than two documents where the scope of related contents becomes more general depending on the co-occurring contents. Applications of document relation discovery include an automatic discovery system of related articles for literature review, an assistant system for article authoring and a novel search engine which takes a set of documents as a query instead of a set of keywords or a document as provided in a conventional method. To discover good document relations, this thesis presents an extension of frequent itemset mining to discover the document relations on an attribute-value database where the values are weighted by real values, instead of boolean values as in the conventional method. The goals of thesis are: (1) to study how well the word-based approach performs in finding relations among documents using frequent itemset mining techniques, (2) to propose a method to automatically evaluate the discovered document relations using a citation graph, and (3) to invent a measure for automatically evaluating the quality of the discovered document relations. The approach is applied to discover word-based relations among scientific publications. The proposed method is evaluated using a set of scientific publications in a digital library to judge the quality of discovered document relations based on their references (citations). With the concept of transitivity as direct/indirect citations, the thesis introduces a series of evaluation criteria, called order accumulative citation matrices, to define the validity (quality) of discovered relations. Two kinds of validity, called soft validity and hard validity, are presented to express the quality of the discovered relations. For the purpose of impartial comparison, the expected validity is statistically estimated based on the generative probability of each document relation pattern. The experimental results show that the discovered document relations using a bigram model as term definition are more valid than those using a unigram model. Stopword removal is a significant scheme for filtering unnecessary terms in the process of representing document content. The results also show that the proposed method successfully discovers a set of document relations, the quality of which is significantly better than its expectation. With the human evaluation of sampled document relations, it is confirmed that the proposed automatic evaluation method based on citation information is a potential approach to evaluate the quality of document relations. Moreover, an extension of the term weighting scheme can enhance the quality of discovered document relations, where inverse document frequency performs well to discover high-valid relations from the collection. Furthermore, the augmented normalized term frequency can help to discover the good quality relations in a higher rank while the bigram term frequency performs well in any rank of discovered document relations.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Nowadays, it has become difficult for researchers to follow the state of the art in their area of interest since the number of research publications has increased continuously and quickly. Such a large volume of information brings about serious hindrance for researchers to position their own works against existing works, or to find useful relations (or connections) between them [Kessler, 1963, Small, 1973, Wilkinson and Smeaton, 1999, Bergmark, 2000, Ganiz et al., 2005]. Although the publication of each work may include a list of related articles (documents) as its reference (called citation), it is still impossible to include all related works due to either intentional reasons (e.g., limitation of paper length) or unintentional reasons (e.g., naïvely unknown). Enormous meaningful connections that permeate the literatures may remain hidden.

As stated in [Hetzler, 1997], there are several types of relations among documents, e.g., attribute-based relations, document-to-document topological relations, and usage-based relations; but the traditional and well-known one, i.e., content-based relations, is focused on in this work. So far several approaches have been proposed to utilize information sources available in the literatures to find these meaningful but unrevealed relations.

Growing from different fields, known as literature-based discovery, the approach of discovering hidden and significant relations within a bibliographic database has become popular in medical-related fields [Swanson, 1986, Swanson, 1990]. As a content-based approach with manual and/or semi-automatic processes, a set of topical words or terms are extracted as concepts and then utilized to find connections between two literatures. Due to the simplicity and practicality of this approach, it was used in several areas in succeeding works [Gordon and Dumais, 1998, Lindsay and Gorden, 1999, Pratt et al., 1999].

As a so-called citation analysis, expansion of bibliography or citation information in scientific publication can be used to find such relations. In the past decades, citation information was proved to be useful for several purposes, including measurement of impact factor [Garfield, 1972], characterization of the citation [Redner, 1998, An et al., 2004], support of browsing citation graph [Lawrence et al., 1999, Chen, 1999] and so forth. For the task of document relation discovery, two basic properties of citation, called bibliographic coupling [Kessler, 1963] and co-citation [Small, 1973], can be focused upon. Several previous works [Egghe and Rousseau, 2002, Garfield, 2001] stated that any two documents tend to have relation with each other if they are citing one or more documents in common (bibliographic coupling) or they are both cited by one or more documents in common (co-citation). Several applications [Nanba et al., 2000, White and McCain, 1989, He and Hui, 2002, Lin et al., 2003, White, 2003, Rousseau and Zuccala, 2004] successfully applied these properties for their tasks. A brief but comprehensive survey on automatic link

generation can be found in [Wilkinson and Smeaton, 1999] However, these works are not fully automated and have a lot of labor intensive tasks.

Besides citation information, words or terms in a document are potential clues for detecting relations between the document and other related documents. Also applied in information retrieval [Salton et al., 1975, Faloutsos and Oard, 1995, Jones and Willett, 1997], text categorization [Nigam et al., 2000, Yang, 1999, Ruch, 2006, Ehrler et al., 2005] and text clustering [da Silva et al., 2001, Beil et al., 2002, Hung and Wermter, 2003], this word-based or term-based approach (later called word-based approach) discovers a set of documents with similar contents (topics) using either word co-occurrences or shared vocabularies. Imitating techniques in information retrieval (IR), a relation between any two documents can be found by means of measuring document similarity. Applying the vector space model originally proposed by Salton et al. [Salton et al., 1975], Furuta et al. [Furuta et al., 1989] presented a comparative study of the quality of links created between two documents or two parts in a document by sharing the glossary. Salton et al. [Salton and Buckley, 1991] described a method to build a set of cross-references for an encyclopedia. Lelu created links using both similarity and spreading activation [Lelu, 1991]. Later, Allan [Allan, 1997] proposed a method that exploits differences between the various sub-types of semantic links, and showed how links associated with these sub-types can be determined and assigned to a pair of documents or two parts in a document. It should be noted that IR-based methods are designed to discover relations between only two documents (binary relations).

Similar to IR, text clustering (TC) discovers a set of similar documents, based on some kind of similarity. However, unlike IR, a discovered document relation may include more than two documents (n-ary relation). A relation can be assumed if documents are assigned to the same group. Although TC looks more general than IR, it still has a few limitations [Glenisson et al., 2003, Ertoz et al., 2003, Moon and Singh, 2005]. First, clustering is designed to deal with a small number of clusters and it is rarely applied to handle a large number of clusters. Second, most clustering methods assume that an object (in this task, a document) belongs to only one cluster. Third, the process is computationally expensive if all potential clusters need to be found in the situation that a document is not limited to only one cluster. This complexity comes from the fact that all document combinations need to be explored for any possible relation.

Moreover, there has been little exploration of how to evaluate document relations discovered from text collections. Most works in text mining utilize a dataset, which includes both queries and their corresponding correct answers, as a test collection. They usually define certain measures and use them for performance assessment on the test collection. For instance, classification accuracy is applied for assessing the class to which a document is assigned in text categorization (TC) [Rosch, 1978], while recall and precision are used to evaluate retrieved documents with regard to given query keywords in information retrieval (IR) [Van Rijsbergen, 1979, Salton and McGill, 1983]. As a more naive evaluation method, human judgment has been used in more recent works on mining web documents, such as HITS [Kleinberg, 1999] and PageRank [Page et al., 1998], where there is no standard dataset. However, this manual evaluation is a labor intensive task and quite subjective.

Compared to TC and IR, the evaluation of discovered document relations is difficult and complicated. For one reason, the process to prepare correct answers in the test collection is labor-intensive with an exponential number of candidate relations (a relation may involve more than two documents) to be evaluated. Moreover, there is a lack of standard criteria for

evaluating document relations. So far, while there have been several benchmark datasets, e.g., UCI Repository[1], WebKB[2], TREC data[3], for TC and IR tasks, there is no standard dataset that is used for this task of document relation discovery.

1.1 Motivations and Goals

The main motivations of this work are to discover the high-quality document relations and present the trustworthy automatic evaluation for assessing those discovered relations. For the first motivation, the work focuses on presenting a novel method that applies frequent itemset mining (FIM) techniques to find n-ary document relations where we can set a minimum support to avoid exploring all document combinations. By encoding documents as items and terms as transactions, each frequent pattern is in the form of a set of documents where its support is introduced by the co-occurring terms. With the frequent itemset mining theory, the preservation of closure properties in the process of candidate generation can help to avoid the exponential number of generating all document combinations. Theoretically, the approach of frequent itemset mining is designed to discover the knowledge on the large-scale databases where most of the studies focuses to fasten the mining process. This advantage brings a good exploration to the document relation discovery where the huge amount of documents is concerned. Although it is possible to apply other approaches, such as clustering technique or information retrieval, for discovering document relations, the processes of those approaches are computational expensive and not mainly focused to find the n-ary document relations.

The latter motivation comes from a suspect of the quality of discovered relations. Although the best way to evaluate the quality of discovered relations is to use human judgment, the task is excessively time-consuming and labor-intensive. Toward resolving these issues, this work also proposes a method to use citation information in research publications as a source for evaluating the discovered document relations. Conceptually, the relations among documents can be formulated as a subgraph where each node represents a document and each arc represents a relation between two documents. Based on this formulation, the transitivity of citation can introduce a huge number of relations between the documents where the documents need not to be directly cited with each other. Those relations can be assumed to be the potential document relations which are indirectly defined by the document's authors, and they can be used as the trust knowledge for the evaluation. A number of scoring methods to measure the validity of discovered relations based on the citation information can be set according to the different decision criteria. Moreover, the results from the human evaluation are also needed to verify the results from the proposed automatic evaluation.

According to the above motivations, three main goals of this work are:

1. To study how well the word-based approach performs in finding relations among documents using frequent itemset mining techniques.

2. To propose a method to automatically evaluate the discovered document relations using a citation graph.

3. To invent a measure for automatic evaluating the quality of the discovered relations.

[1] http://www.ics.uci.edu/∼mlearn/MLRepository.html
[2] http://www.webkb.org/
[3] http://trec.nist.gov/data.html

1.2 Contributions

There are several contributions in this research as follows.

1. *An efficient method for document relation discovery*. Using the notion of encoding database as items and terms as transactions, an efficient approach of extended frequent itemset mining is proposed to mine the frequent patterns with the modification of original support definition. Those frequent patterns are assumed to be the document relations where the relations are introduced by the co-occurring terms.

2. *An analysis of document representation that is suitable for document relation discovery*. Since there are several schemes to define the terms in the documents, the investigation on document representation is needed for selecting the suitable term definition and term weighting schemes to well represent the document contents. With the good document contents, the high-quality document relations can be discovered.

3. *A formulation of citation graph as the n-ary relations between the documents*. The citations between documents can be formulated as the citation graph. Furthermore, the transitive of citations can introduce a huge number of relations between the documents where the documents need not to be directly cited with each other. Those relations are assumed to be the potential document relations which are indirectly defined by the document's authors, and they can be used as the benchmark.

4. *A trustworthy method for automatic evaluation based on the citation information to judge the quality of discovered document relations*. It is a labor-intensive and time-consuming task to evaluate a large set of document relations by hands, therefore an automatic evaluation is a promising way to validate the approach of document relation discovery when the document collections are not specific to be only the standard corpus. With the notion of citation formulation, the automatic evaluation is proposed to validate the discovered document relations in both soft and hard decisions. The trustworthy of the proposed evaluation is also verified by the consistence of the results with human evaluation.

5. *A set of document relations that can be applied to several potential applications*.

For the application, the document association networks which reveal the relations among documents or groups of documents where the relations are specified by the labels that bind those documents by their common coincident information can be constructed. The document association network can be illustrated by graph visualization as shown in Figure 1.1. Note that only the approach to construct the relations among documents and groups of documents is focused on in this work, the method to label the relations is not taken into consideration. This novel representation has various contributions to many fields. For example in IR, rather than representing the search result by a list of individual documents, it can be shown by the document association networks of those retrieved documents which facilitate in browsing the desired information. Together with the social document network approach, the associations among the documents in the same social network (or even across social networks) can be discovered to produce extensive knowledge.

Moreover, several applications can be implemented using the method of document relation discovery. Here are the examples of such applications.

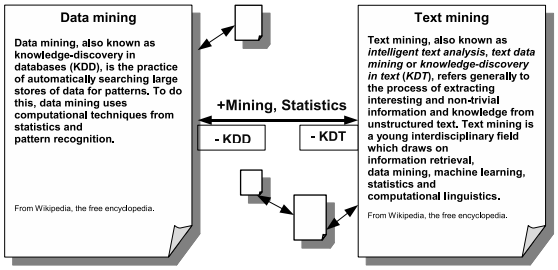


Figure 1.1 An example of document relations

- Automatic related article discovery for literature review and assistant system for article authoring
- Discovering novel connections or knowledge among similar/different research areas
- Duplicate article detection for publication review system
- Novel search engine when the given query is a set of documents (not only keywords or a document)

1.3 Thesis Structure

Chapter 2 - *Background* reviews the current state-of-the-art in both document relation discovery and utilizing citation graph as information for extracting relations.

Chapter 3 - *Discovery of Document Relations* presents a method for discovering document relations using frequent itemset mining. By encoding documents as items, and terms in the documents as transactions, a frequent itemset that we can find will be in the form of a set of documents which share a large number of terms. To represent documents in the database, several combinations of term definition and term weighting schemes are explored as the parameters for extracting high-quality document relations.

Chapter 4 - *Evaluation of Document Relations: An Automatic Evaluation* proposes a method to use citation information in research publications as a source for automatically evaluating the discovered document relations. a series of measures called v-validity is defined on direct/indirect citations formulated by so-called order accumulative citation matrices. Moreover, this work proposes generative probability that is derived from probability theory and uses it to compute an expected score to capture objectively how good evaluation results are.

Chapter 5 - *Experimental Results and Evaluations* shows the evaluation results on various document representations including the comparison with the statistical generative probability. However, the results from automatic evaluation are also compared with the results from human evaluation to confirm the potential of the proposed evaluation method.

Chapter 6 - *Experimental Results on Various Term Weighting* investigates several extensions of applying term weighting schemes to enhance the quality of discovered document relations.

Chapter 7 - *Conclusions and Future Work* contains a summary of the work covered and conclusions reached. Potential enhancements and future research are discussed.

Chapter 2

Background

This chapter presents some background which is related to various approaches for document relation discovery. The topics include several characteristics of relations discovered from a set of documents especially on the scientific publications. Besides these topics, some related works on evaluating the document relations are also surveyed and discussed. Since the frequent itemset mining will be used as a method for document relation discovery, a background of frequent itemset mining and the traditional problem of association rule mining is presented in the last section. Two algorithms, Apriori and FP-Tree, are described in more detail including a discussion about their performances. Since the efficient algorithm, FP-Tree, will be used as a main method to discover document relations in this work, its advantage and disadvantage are also discussed at the end of this chapter.

2.1 Related Works on Document Relation Discovery

As a citation-based approach, Lawrence et al. [Lawrence et al., 1999] proposed a similarity measure, called CCIDF, based on common citations to judge the relatedness between articles. Imitating TFIDF in the text-based similarity, the CCIDF corresponds to the multiplication of the number of common citation and inverse document frequency. The CCIDF metric is used by the automatic citation indexing system in the Citeseer. Recently, Rahal et al. [Rahal et al., 2006] have proposed a method to discover research trends (subject-matter history, extensions or evolution over time) by analysing semantics hidden in the edges of a citation graph using association rule mining. By modeling an edge in the citation graph as a transaction whereas the subjects of the citee and the subjects of the citer are the items in such a transaction, a set of association rules are mined.

As stated in [Lu et al., 2006], use of citation information to compute relatedness between scientific papers has been studied in the well-known work for citation indexes [Garfield, 1995]. Since citations of other papers are hand-picked by the authors as being related to their research, the reference list of a paper contains information which can be exploited to judge relatedness. The simplest relation, a direct reference or citation, is likely to occur among related papers which are published apart in time. It does not occur very frequently among papers published in the same year or very close in time. Two different citation relations between papers have been specifically identified and used to calculate similarity, namely co-citation (two papers referenced by the same paper) and bibliographic coupling (two papers citing the same paper) [Small, 1973]. Two papers are related by co-citation if they are cited together by the same paper. Small has studied the co-citation pattern among research papers and highlights its importance in similarity computation. Co-citation links are often present

in two related older papers. Two papers are bibliographically coupled, if they reference the same paper. If two recent papers are published in the same or similar research area, a bibliographic coupling pattern is very likely to be found in their reference lists. Bibliographic coupling and co-citation have been employed to compute similarity between research papers. But each of them is only suitable for computing similarity in specific cases. For instance, researchers have used co-citation frequency to compute relatedness between two papers, but the papers to be judged have to be well cited by other authors for the algorithm to work properly. Apparently co-citation is not efficient in judging similarity among recent papers which have not yet had the chance to be cited by many other authors. In terms of the direct link pattern, if the two papers are published almost at the same time, a direct citation link is not likely to be found between them, even if their content is related. Similarly, papers which appeared in the early stages of the development of a research specialty are not good candidates for bibliographic coupling analysis. In our metrics, we do not need to know which of these citation patterns our papers fall under. All patterns of citation relations are accounted for by using the citation graph.

As the combination of citation-based and word-based approaches, some works utilized both citation and word/term information [Wilkinson and Smeaton, 1999, Nanba et al., 2000]. Kostoff et al. [Kostoff et al., 2001] introduced an approach to combine citation bibliometrics and text mining (i.e., categorization) for analyzing the impact of an originating research article on other citing research/application over time along with the pathways through the achievement. As a more recent work, Lu and his colleague [Lu et al., 2006] proposed a method to use information of local neighborhood articles of an article to calculate the similarity between two articles that share some citations under the concept of transitivity, which is extensively studied in [Bjorneborn, 2004]. Although the citation-based similarity was its main focus, the work compared its result with that of the word-based similarity. It was concluded that the word-based approach outperformed the citation-based similarity to some extent. While most existing works on linking related documents occupied the citation-based approach, very few works explored the word-based approach that utilizes the content in the articles to determine the similarity.

Growing from different research field, known as literature-based discovery, the approach of discovering hidden and significant relations within a bibliographic database has been popular in medical-related fields since 1986 [Swanson, 1986, Swanson, 1990]. As a word-based approach with manual and/or semi-automatic process, a set of topical words or terms are extracted as concepts and then utilized to find a connection among two separate arguments. Due to the simplicity and practicality of this approach, it was used in several areas by its succeeding works [Gordon and Dumais, 1998, Lindsay and Gorden, 1999, Pratt et al., 1999]. So far, although there have been several works on mining relations among concepts in documents, very few attempts are made to fully automate the process of discovering relations at the level of documents by exploiting their content words or terms.

2.2 Related Works on Evaluation of Document Relations

The other important issue is related to how to evaluate the discovered results. In general, the obtained relations can be evaluated based on either subjective measures such as consistency to human answers [Lu et al., 2006, Padmanabhan and Tuzhilin, 1999, Silberschatz and Tuzhilin, 1995, Silberschatz and Tuzhilin, 1996] or objective measures such as interestingness, leverage and

conviction [Rahal et al., 2006, Klemettinen et al., 1994]. While the subjective evaluation needs a labor-intensive task to provide answers by human, the objective evaluation may not reflect how much the relations match with human intuition (belief). The background of those evaluations are reviewed as follows.

In the area of data mining, the evaluation which is usually applied to evaluate the discovered knowledge is to use the interestingness measures. These measures are calculated based on the statistical significance of discovered knowledge to the dataset itself or novelty of knowledge to the human intuition. As described in [Rahal et al., 2006], interestingness measures for the discovered knowledge can fall in one of two classes : objective measures and subjective measures. Objective interestingness measures are data-centric in that they define the interestingness of a pattern in terms of the data used in the mining process. They also highly depend on the structure of the patterns. For example, in ARM, the two ubiquitous objective measures are support and confidence, both of which highlight statistical properties relating to the discovered rules. Due to the many complexities arising in the pattern discovery process, objective measures usually discover a large number of patterns and thus fall short of their purpose (to discover useful and comprehensible knowledge from huge amounts of data), especially when the notion of interestingness depends on additional factors such as the decision-maker.

A number of subjective measures [Padmanabhan and Tuzhilin, 1999, Silberschatz and Tuzhilin, 1995, Silberschatz and Tuzhilin, 1996] have been proposed for the above scenario. In general, subjective measures endeavor to generate a smaller tailored set of patterns that is potentially more interesting and useful to the pattern examiner. As discussed by Silberschatz and Tuzhilin [Silberschatz and Tuzhilin, 1996], subjective measures depend on two main factors to discover patterns, namely, actionability and unexpectedness. Actionability states that a pattern is considered interesting to the examiner if it calls for action on his or her behalf. Unexpectedness focuses more on the surprising factor of the pattern with respect to the examiner (i.e., the degree to which the pattern surprises the examiner). In order for the unexpectedness factor to be integrated into a subjective measure, a system of beliefs [Silberschatz and Tuzhilin, 1996] must be defined first. Such a system would define the standard knowledge expected by the examiner. The discovery process then captures all deviations from such standards as unexpected and thus as interesting to the examiner.

In general, beliefs can be either hard or soft. Hard beliefs represent the knowledge that the examiner is not willing to change even in the light of newly discovered contradictory evidence; the validity of the discovered patterns and sometimes of the original data is questioned instead. On the other hand, soft beliefs could be changed by the examiner if suggested by new patterns. A user-defined measure of strength, referred to as the degree of belief, is usually associated with every belief in the system. A number of subjective interestingness measures for association rules are presented next. Pietracaprina and Zandolin [Pietracaprina and Zandolin, 2003] use a probabilistic approach to discover unexpected rules in the form of rule-pairs. Their work is domain independent in that it requires no prior knowledge in the form of beliefs against which the unexpectedness factor is measured. In order to arrive at the subset of interesting rules, the authors assume the validity of what they refer to as the monotonicity of beliefs which states that if a belief holds on some data with some degree then it must also hold on large subsets of that data. However, in those works, it is a question to set the beliefs that are reasonable and general enough for evaluating the discovered knowledge in the domain of document relations.

In the work of literature-based discovery, the approach to evaluate the discovered knowledge from literature-based discovery is not easy as stated in [Ganiz et al., 2005]. It is a multi-faceted task which requires human judgment and needs some periods of evaluation time. The systems pose an additional fundamental challenge in evaluation because, if they are successful, then by definition they are capturing new knowledge that has yet to be proven useful [Pratt and Yetisgen-Yildiz, 2003]. Evidence supporting the preliminary discoveries of Swanson [Swanson, 1986] was provided later by medical researchers after the initial discoveries were made by trial and error [Gordon and Lindsay, 1996]. Providing evidence in support of such discoveries is only one perspective of evaluation. Evaluation can also be based on the generated results. The correctness of the results returned, often measured as accuracy and/or precision, is one such metric. Recall is another, which refers to the number of correct results returned compared to the total number of correct results available. Other metrics reflect more qualitative aspects of the system, such as complexity of the user interface. For systems whose aim is to support human experts in the discovery process, usability issues are very important. Finally, the human experts role is still an important evaluation method for the literature-based discovery systems.

With the more related works on IR, several standard datasets are available to test the proposed techniques. They include both training data and testing data, including the correct answers with respect to the query as a test collection. They define the performance of the search strategies through two main measures, i.e., precision and recall. The precision presents the proportion of retrieved and relevant documents to all the documents retrieved, while the recall indicates the proportion of relevant documents that are retrieved out of all relevant documents available. [Van Rijsbergen, 1979, Salton and McGill, 1983].

However, there are some drawbacks of the traditional evaluation method that uses precision and recall as stated in [Jin et al., 2001]. First, it requires collecting relevance judgments of human subjects for every document to every query, which is very expensive and time consuming because of the large size of the text collection. Even though, all TRECs use the sampling technology [Jones and van Rijsbergen, 1975], i.e., only the union set of the top 100 retrieved documents from different search systems will be accessed by human subjects, it still needs a large amount of human effort. Second, it relies on human relevance judgment, which usually is very subjective. It is well known that, quite often people can have different opinions on whether a document is relevant to a query [Mizzaro, 1999]. Thus, evaluation of information retrieval systems based on human relevance judgments may be biased by the human subjects and does not reflect the true performance of systems. Therefore, there is an attempt in [Jin et al., 2001] to bypass the need for human judgments to evaluate the quality of the term weighting models used in IR systems. In such work, the meta-scoring scheme was proposed to judge the goodness of term weightings by analyzing the document vectors. The measure is quite subjective and heavily depends on the characteristic of the dataset.

To this end, we try to model the automatic evaluation which can be applied to evaluate the discovered document relations by using trustworthy information. Since there are several approaches that try to utilize the citation graph for document relation discovery, it is interesting to take the citation graph into consideration for formulating the evaluation on document relations discovered from the word-based approach. This will be investigated again in a later chapter.

2.3 Related Works on Frequent Itemset Mining

In this section, the traditional association rule mining problem will be first introduced and the frequent itemset mining algorithms will be then described and discussed.

2.3.1 Association Rule Mining

The formal statement of association rule mining was firstly stated in [Agrawal et al., 1993a] by Agrawal. Let $I = I_1, I_2, ..., I_m$ be a set of m distinct attributes, T be transaction that contains a set of items such that $T \subseteq I, D$ be a database with different transaction records Ts. An association rule is an implication in the form of $X \rightarrow Y$, where $X, Y \subseteq I$ are sets of items called itemsets, and $X \cap Y = \phi$. X is called antecedent while Y is called consequent, the rule means X implies Y. There are two important basic measures for association rules, support(sup) and confidence(conf). Since the database is large and users are concerned about only those frequently occurring items, usually thresholds of support and confidence are pre-defined by users to drop those rules that are not so interesting or useful. The two thresholds are called minimum support (minsup) and minimum confidence (minconf), respectively. Additional constraints of interesting rules also can be specified by the users. The two basic parameters of Association Rule Mining (ARM) are: support and confidence.

Support of an association rule is defined as the percentage/fraction of records that contain $X \cup Y$ to the total number of records in the database. The count for each item is increased by one every time the item is encountered in different transaction T in database D during the scanning process. It means the support count does not take the quantity of the item into account. For example in a transaction a customer buys three bottles of beers but we only increase the support count number of beer by one; in other words, if a transaction contains an item then the support count of this item is increased by one. Support is calculated by the following formula:

$$Support(X \cup Y) = \frac{\text{the number of transactions which contain } X \cup Y}{|T|}$$

From the definition we can see, support of an item is a statistical significance of an association rule. If the support of an item is 0.1%, it means only 0.1 percent of the transactions contain purchasing of this item. The retailer will not pay much attention to such kind of items that are not bought so frequently. Obviously a high support is desired for more interesting association rules. Before the mining process, users can specify the minimum support as a threshold, which means they are only interested in certain association rules that are generated from those itemsets whose supports exceed that threshold. However, sometimes even though the itemsets are not so frequent as defined by the threshold, the association rules generated from them are still important. For example, in the supermarket some items are very expensive, consequently they are not purchased so often as the threshold required, but association rules between those expensive items are as important as other frequently bought items to the retailer.

Confidence of an association rule is defined as the percentage/fraction of the number of transactions that contain $X \cup Y$ to the total number of records that contain X, where if the percentage exceeds the threshold of confidence an interesting association rule $X \rightarrow Y$ can be generated.

$$Confidence(X \rightarrow Y) = \frac{Support(X \cup Y)}{Support(X)}$$

Confidence is a measure of strength of the association rules, suppose the confidence of the association rule $X \rightarrow Y$ is 80%. This means that 80% of the transactions that contain X also contain Y. Similar to ensuring the interestingness of the rules, specified minimum confidence is also pre-defined by users.

Association rule mining is to find out association rules that satisfy the pre-defined minimum support and confidence from a given database [Agrawal and Srikant, 1994]. The problem is usually decomposed into two subproblems. One is to find those itemsets whose occurrences exceed a predefined threshold in the database, those itemsets are called frequent or large itemsets. The second problem is to generate association rules from those large itemsets with the constraints of minimal confidence. Suppose one of the large itemsets is $L_k, L_k = I_1, I_2, ..., I_{k-1}, I_k$. Association rules with this itemset are generated in the following way: the first rule is $I_1, I_2, ..., I_{k-1} \rightarrow I_k$, and by checking the confidence this rule can be determined as interesting or not. Then other rules are generated by deleting the last item in the antecedent and inserting it to the consequent, further the confidences of the new rules are checked to determine the interestingness of them. Those processes iterate until the antecedent becomes empty. Since the second subproblem is quite straight forward, most of the researches focus on the first subproblem.

2.3.2 Traditional Approach

Since association rule mining is a well-explored research area, we will only introduce some basic and classic approaches for association rule mining. As stated before, the second subproblem of ARM is straightforward, most of those approaches focus on the first subproblem (frequent itemset mining; FIM). The first subproblem can be further divided into two subproblems: candidate large itemsets generation process and frequent itemsets generation process. We call those itemsets whose support exceed the support threshold, large or frequent itemsets. Those itemsets that are expected or have the hope to be large or frequent are called candidate itemsets. Most of the algorithms of mining association rules we surveyed are quite similar, the difference is the extent to which certain improvements have been made, so only some of the milestones of association rule mining algorithms will be introduced. First we will introduce some naive and basic algorithms for association rule mining, Apriori series approaches. Then another milestone, tree structured approaches, will be explained. Finally, this section will end with some special issues of association rule mining, including multiple level ARM, multiple dimension ARM, constraint based ARM and incremental ARM. In order to make it easier for us to compare those algorithms we use the same transaction database, a transaction database from a supermarket, to explain how those algorithms work. This database records the purchasing attributes of its customers. Suppose during the pre-processing step all those attributes that are not relevant or useful to our mining task are pruned, only those useful attributes are left ready for mining as shown in figure 2.1.

2.3.3 Frequent Itemset Mining Algorithms

In this section, the original Apriori algorithm is introduced to give a clear view of the frequent itemset mining process. However, this algorithm is not efficient enough to deal with

TID	List of items
T100	I_1,I_2, I_5
T200	I_2, I_4
T300	I_2, I_3
T400	I_1, I_2, I_4
T500	I_1, I_3
T600	I_2, I_3
T700	I_1, I_3
T800	I_1, I_2, I_3, I_5
T900	I_1, I_2, I_3
T000	I_1, I_2, I_5, I_6

Figure 2.1 An example of original databases

large databases. Therefore, another more efficient algorithm called FP-Tree is presented and will be used as an algorithm for document relation discovery in this work. To this end, their performances are also described and discussed.

Items	Count number
I_1	7
I_2	8
I_3	6
I_4	2
I_5	3
I_6	1

(a) C_1

Large 1 Items
I_1
I_2
I_3
I_5

(b) L_1

Items	Count number
I_1, I_2	5
I_1, I_3	4
I_1, I_5	3
I_2, I_3	4
I_2, I_5	3
I_3, I_5	1

(c) C_2

Large 2 Items
I_1, I_2
I_1, I_5
I_2, I_5
I_2, I_3
I_1, I_3

(d) L_2

Items	Count number
I_1, I_2, I_5	3
I_1, I_2, I_3	2

(e) C_3

Figure 2.2 Apriori mining process

Apriori Algorithm

The Apriori algorithm, a great improvement in the history of association rule mining, was first proposed by Agrawal in [Agrawal and Srikant, 1994]. The AIS [Agrawal et al., 1993a] is just a straightforward approach that requires many passes over the database, generating many candidate itemsets and storing counters of each candidate while most of them turn out to be not frequent. Apriori is more efficient during the candidate generation process for two reasons: Apriori employs a different candidates generation method and a new pruning technique. There are two processes to find out all the large itemsets from the database in the Apriori algorithm. First the candidate itemsets are generated, then the database is scanned to check the actual support count of the corresponding itemsets. During the first scanning of the database the support count of each item is calculated and the large 1-itemsets are generated by pruning those itemsets whose supports are below the pre-defined threshold as shown in Figure 2.2(a) and (b). In each pass only those candidate itemsets that include the same specified number of items are generated and checked. The candidate *k*-itemsets

```
Input:
        database D
        Mini Support ε
        Mini Confidence ξ
Output:
        R_t All association rules
Method:
01      L_1 = large 1-itemsets;
02      for(k=2; L_{k-1} ≠ Ø; k++) do begin
03      C_k =apriori-gen(L_{k-1}); //generate new candidates from L_{k-1}
04      for all transactions T ∈ D do begin
05      C_t=subset(C_k,T); //candidates contained in T.
06      for all candidates C ∈ C_t do
07      Count(C)=Count(C)+1; // increase support count of C by 1
08      end
09      L_k={C ∈ C_t | Count(C) ≥ ε × | D | }
10      end
11      L_f= ⋃_k L_k;
12      R_t=GenerateRules(L_f, ξ )
```

Figure 2.3 Apriori algorithm

are generated after the $(k-1)^{th}$ passes over the database by joining the frequent $k-1$-itemsets. All the candidate k-itemsets are pruned by checking their sub $(k-1)$-itemsets. If any of its sub $(k-1)$-itemsets are not in the list of frequent $(k-1)$-itemsets, this k-itemsets candidate is pruned out because it has no hope to be frequent according to the Apriori property. The Apriori property says that every sub $(k-1)$-itemsets of the frequent k-itemsets must be frequent. Let us take the generation of candidate 3-itemsets as an example. First all the candidate itemsets are generated by joining frequent 2-itemsets, which include $(I_1,I_2,I_5),(I_1,I_2,I_3),(I_2,I_3,I_5),(I_1,I_3,I_5)$. Those itemsets are then checked for their sub itemsets, since (I_3,I_5) is not frequent 2-itemsets, the last two 3-itemsets are eliminated from the list of candidate 3-itemsets as shown in Figure 2.2(e). All those processes are executed iteratively to find all frequent itemsets until the candidates itemsets or the frequent itemsets become empty. The result is the same as the AIS algorithm. The algorithm is shown in Figure 2.3. In the process of finding frequent itemsets, Apriori avoids the wasted effort of counting the candidate itemsets that are known to be infrequent. The candidates are generated by joining the frequent itemsets level-wisely and candidates are pruned according to the Apriori property. As a result, the number of remaining candidate itemsets ready for further support checking becomes much smaller, which dramatically reduces the computation, I/O cost and memory requirement. Details of the Apriori-gen and GenerateRules functions were elaborated in [Agrawal and Srikant, 1994]. The Apriori algorithm still inherits the drawback of scanning whole databases many times. Based on the Apriori algorithm, many new algorithms were designed with some modifications or improvements. Generally there were two approaches: one was to reduce the number of passes over the whole database or replacing the whole database with only part of it based on the current frequent itemsets, another approach was to explore different kinds of pruning techniques to make the number of candidate itemsets much smaller. Apriori-TID and Apriori-Hybrid [Agrawal and Srikant, 1994] , DHP [Park et al. 1995], SON [Savesere et al. 1995] are modifications of the Apriori algorithm. Most of the algorithms introduced above are based on the Apriori algorithm and try to improve the efficiency by making some modifications, such as reducing the number of passes over the database; reducing the size of the database to be scanned in every pass; pruning

the candidates by different techniques and using sampling technique. However there are two bottlenecks of the Apriori algorithm. One is the complex candidate generation process that uses most of the time, space and memory. Another bottleneck is the multiple scan of the database.

FP-Tree (Frequent Pattern Tree) Algorithm

To break the two bottlenecks of Apriori series algorithms, some works of association rule mining using tree structure have been designed. FP-Tree [Han et al., 2000, Han et al., 2004], frequent pattern mining, is another milestone in the development of association rule mining, which breaks the two bottlenecks of the Apriori. The frequent itemsets are generated with only two passes over the database and without any candidate generation process. FP-Tree was introduced by Han et al in [Han et al., 2000] and [Han et al., 2004]. By avoiding the candidate generation process and making fewer passes over the database, FP-Tree is an order of magnitude faster than the Apriori algorithm. The frequent patterns generation process includes two subprocesses: constructing the FT-Tree, and generating frequent patterns from the FP-Tree. The process of constructing the FP-Tree is as follows.

(1) The database is scanned for the first time, during which the support counts of each item are collected. As a result the frequent 1-itemsets are generated as shown in Figure 2.4(b). This process is the same as in the Apriori algorithm. Those frequent itemsets are sorted in descending order of their supports and the head table of ordered frequent 1-itemsets is created as shown in Figure 2.5.

(2) Create the root node of the FP-Tree T with a label of *Root*. The database is scanned again to construct the FP-Tree with the head table, for each transaction the order of frequent items is resorted according to the head table. For example, the first transaction (I_1, I_2, I_5) is transformed to (I_2, I_1, I_5), since I_2 occurs more frequently than I_1 in the database. Let the items in the transaction be [p | P], where p is the most frequent item and P is the remaining items list, and call the function *Insert*[p | P]; T.

(3) The function *Insert*[p | P]; T works as follows. If T has a child N such that N.item-name=p.item-name then the count of N is increased by 1, else a new node N is created and N.item-name=p.item-name with a support count of 1. Its parent link is linked to T and its node link is linked to the node with the same item-name via a sub-link. This function *Insert*P;T is called recursively until P becomes empty.

Let's take the insertion of the first transaction to the FP-Tree as an example to illustrate the insert function and construction of FP-Tree we mentioned above. After reordering this transaction is (I_2, I_1, I_5), so p is I_2 in this case, while P is (I_1, I_5). Then we call the function of insert. First we search and determine whether the node I_2 exists in the tree or not and it turns out I_2 is a new node. According to the rules, a new node named I_2 is created with a support count of 1. Since here T is *Root*, node I_2 is linked to *Root* and we call the insert function again. At this time p is I_1, P is I_5, T is I_2. The result of the FP-Tree of the database is shown in Figure 2.5.

The frequent patterns are generated from the FP-Tree by the procedure named FP-growth [Han et al., 2000, Han et al., 2004]. Based on the head table and the FP-Tree, frequent patterns can be generated easily. It works as shown in Figure 2.6. For example, here is the whole process of getting all those frequent itemsets concerning I_5. Following the head table, we find the pattern base of this node, which are all those paths which end with this node. For

TID	List of items
T100	I_1,I_2, I_5
T200	I_2, I_4
T300	I_2, I_3
T400	I_1, I_2, I_4
T500	I_1, I_3
T600	I_2, I_3
T700	I_1, I_3
T800	I_1, I_2, I_3, I_5
T900	I_1, I_2, I_3
T000	I_1, I_2, I_5, I_6

(a)
Original Database

Large 1 Items	Support
I_1	7
I_2	8
I_3	6
I_5	3

(b) L_1

TID	Ordered Large Items
T100	I_2, I_1, I_5
T200	I_2
T300	I_2, I_3
T400	I_2, I_1
T500	I_1, I_3
T600	I_2, I_3
...	...

(c) Transformed Data

Figure 2.4 FP-Tree data transformation

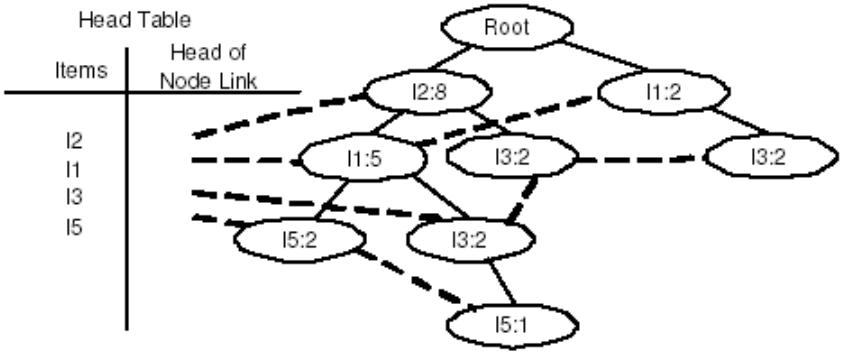


Figure 2.5 Result of FP-Tree

I_5, its pattern base is: $(I_2,I_1)(2)$ and $(I_2,I_1,I_3)(1)$, the number in the bracket following the itemsets means the support of this pattern. Then the count of all the items in the pattern base are accumulated, in this case we get $I_2(3)$, $I_1(3)$ and $I_3(1)$. By checking the support count with the minimal support threshold, the conditional FP-Tree of I_5 is generated $(I_2,I_1)(3)$. Consequently we generate the frequent itemsets/pattern (I_2,I_1,I_5). The mining result is the same with Apriori series algorithms.

The efficiency of FP-Tree algorithm is based on three reasons. First, the FP-Tree is a compressed representation of the original database because only frequent items are used to construct the tree, other irrelevant information is pruned. Also, by ordering the items according to their supports, the overlapping parts appear only once with different support counts. Secondly, this algorithm only scans the database twice. The frequent patterns are generated by the FP-growth procedure. Constructing the conditional FP-Tree which contains patterns with specified suffix patterns, frequent patterns can be easily generated as shown in above the example. Also the computation cost decreased dramatically. Thirdly, FP-Tree uses a divide and conquer method that considerably reduced the size of the subsequent conditional FP-Tree, longer frequent patterns are generated by adding a suffix to the shorter frequent patterns. In [Han et al., 2000] and [Han et al., 2004], there are examples to illustrate all the details of this mining process. Every algorithm has its limitations. The FP-Tree is difficult

```
Input:
     the FP-Tree Tree
Output:
     R_t Complete set of frequent patterns
Method:  Call FP_growth(Tree , null).
Procedure FP-growth (Tree , α)
{
01     if Tree contains a single path P
02     then for each combination (denoted as β) of
       the nodes in the path P do
03     generate pattern β∪α with
       support = minimum support of nodes in β;
04     else for each a_i in the header of Tree do {
05     generate pattern β = a_i∪α with
       support = a_i· support;
06     construct β's conditional pattern base and then
       β's conditional FP-tree Tree_β;
07     if Tree_β ≠ ϕ
08     then call FP-growth (Tree , β)     }
}
```

Figure 2.6 FP-Tree algorithm

to use in an interactive mining system. During the interactive mining process, users may change the threshold of support according to the rules. However for FP-Tree the changing of support may lead to repetition of the whole mining process. Another limitation is that FP-Tree is not suitable for incremental mining. As time goes on databases keep changing and new datasets may be inserted into the database. Those insertions may also lead to a repetition of the whole process if we employ the FP-Tree algorithm.

2.4 Related Works on Generalized Frequent Itemset Mining

With the original approach of association rules mining, the discovered knowledge may not provide desired knowledge in the database. It may be limited with the granularity over the items. For example, a rule "5% of customers who buy wheat breads, also buy chocolate milk" is less expressive and less useful than a more general rule "30% of customers who buy bread, also buy milk". For this reason, generalized association rule mining (GARM) was developed using the information of a pre-defined taxonomy over the items. The taxonomy is a piece of knowledge, e.g., the classification of the products (or items) into brands, categories, product groups, and so forth. Given a taxonomy where only leaf nodes (leaf items) are presented in the transactional database, more informative, initiative and flexible rules (called generalized association rules) can be mined from the database. Each generalized association rule contains items from any level of a taxonomy. Similar to ARM, the most important problem of GARM is how to efficiently find all generalized frequent itemsets, which is the computational intensive step.

In the past, there were still few works related to GARM. Most of them focus on the performance improvement to mine generalized frequent itemsets. In [Srikant and Agrawal, 1997], five algorithms named Basic, Cumulate, Stratify, Estimate and EstMerge were proposed. These algorithms apply the horizontal database and breath-first search strategy like Apriori-based algorithms [Agrawal and Srikant, 1994]. They use the extended database, constructed by adding all distinct ancestors of each item existing in its original transaction, to mine all generalized frequent itemsets. Most methods in GARM exploit some constraints among itemsets for pruning, and discarding meaningless itemsets, i.e., the itemsets containing both an item and its ancestor according to the given taxonomy. However, these algorithms waste a lot of time in multiple scanning of the database even if the sampling method is applied. As a more efficient algorithm, Prutax [Hipp et al., 1998] applies a so-called vertical database format to reduce the computational time needed for multiple scanning of the database. Instead of "generate and test" as done in previous algorithms, it avoids generating meaningless itemsets by using hash tree checking. Nevertheless, in Prutax the limitation is the cost of checking whether their ancestor itemsets are frequent or not by using hash tree before counting their actual support. There exists a slightly different task for dealing with multiple different minimum support in different levels of itemsets as shown in [Han and Fu, 1999] and [Lui and Chung, 2000]. A parallel algorithm has also been proposed in [Shintani and Kitsuregawa, 1998]. Some recent applications that utilize a GARM are shown in [Michail, 2000] and [Hwang and Lim, 2002]. Our preliminary research related to GARM is shown in [Sriphaew and Theeramunkong, 2002].

In this work, we also introduce a new approach for efficiently finding all generalized frequent itemsets using two types of constraints on two generalized itemset relationships, called *subset-superset* and *ancestor-descendant*. We show that it is sufficient to mine only a small set of generalized closed frequent itemsets instead of mining a large set of conventional generalized frequent itemsets. Two algorithms, named *SET* and *cSET*, are proposed to efficiently find generalized frequent itemsets and generalized closed frequent itemsets, respectively. The details of the algorithms are given in Appendix A.

Although the algorithms of generalized frequent itemset mining are not applied to discover the document relations in this work, it is a promising approach that can be used to discover the document relations where the relations express on different granularity of documents. The problem of document relation discovery can be viewed as a problem of generalized frequent itemset mining, where the documents are partitioned into small portions or grouped as the classes and we can find the relations among those portions of documents or classes. However, this exploration has several parameters and is far beyond the scope of this thesis; it is then left as the future work.

Chapter 3

Discovery of Document Relations

In the past, association rule mining (ARM) was well-known as a process to find frequent co-occurrences (frequent patterns) and high confidence if-then rules (association rules) in a database [Agrawal et al., 1993b]. As a prominent technique in data mining, it is useful in various applications such as market basket analysis, fraud detection, data classification, etc. In the ARM process, frequent itemset mining (FIM) is the most essential task to find frequently occurring itemsets from a transactional database. In general, the conventional transactional database is presented in the term of item existences in the transaction. Although most FIM works deal with this kind of database, there are some attempts to extend the original framework to be able to assign the weights for items or transactions in the database, called weighted association rule mining [Cai et al., 1998, Tao et al., 2003, Yun and Leggett, 2006]. In those works, items or transactions are independently weighted with regard to which type of discovered rules we would like to find. The higher weighted items or transactions will obtain higher priority for user interests. However, this approach gives a fixed weight to each item regardless of the transaction in which such item occurs. Then, it does not match with the application where the weight of an item also depends on the transaction in which it exists. This chapter introduces a more general concept of frequent itemset mining which extends from the original FIM to mine frequent itemsets on a database with weighted item-transaction values. For clarity, the concept is explained using an example that matches with our purpose to find document relations.

3.1 Extended Frequent Itemset Mining for Document Relation Discovery

document	terms
d_1	t_1, t_2, t_3, t_4
d_2	t_1, t_2, t_3, t_4
d_3	t_2, t_3
d_4	t_2, t_3, t_4

term	documents
t_1	d_1, d_2
t_2	d_1, d_2, d_3, d_4
t_3	d_1, d_2, d_3, d_4
t_4	d_1, d_2, d_4

Figure 3.1 Document-term orientation (left) and term-document orientation (right)

Figure 3.1 shows two possible representations of a database: the document-term and the term-document orientations. Using different orientations of an attribute-value database as an input for FIM, different kinds of knowledge will be discovered. For the document-term orientation, the discovered frequent itemset is a set of highly co-occurring terms in the documents. Based on this, some text mining approaches were proposed to extract related terms

	d_1	d_2	d_3	d_4
t_1	1	1	0	0
t_2	1	1	1	1
t_3	1	1	1	1
t_4	1	1	0	1

	d_1	d_2	d_3	d_4
t_1	4	2	0	0
t_2	4	2	4	1
t_3	2	4	2	2
t_4	2	4	0	1

Figure 3.2 Boolean-valued (left) and real-valued (right) databases

from a set of documents [Feldman et al., 1998, Clifton and Cooley, 1999, Nahm and Mooney, 2000, Theeramunkong, 2004]. For the term-document orientation, the discovered frequent itemset is prominently changed to be a set of documents which share a large number of terms. The discovered results can be assumed as a word-based relation among documents where the relation is introduced by the coincident terms. This point is originally focused upon in this thesis. A transactional database with boolean (binary) values is generalized to that of any real (non-binary) values. Figure 3.2 shows two alternative attribute-value databases; the boolean-valued and the real-valued databases. Here, a real value indicates a weight of an attribute (item) in the transaction, e.g., a function of how often the attribute appears in the transaction, or (perhaps) the relative frequency of that attribute in the overall set of transactions. In the field of text processing, the weight can be defined in the form of vector space model (VSM), introduced by Salton [Salton et al., 1975]. In this case, such weight is defined by a so-called term frequency of a term in the document. Note that in this work, a transaction corresponds to a term while an item corresponds to a document. Therefore, a "docset" (document set) is used in place of the term "itemset", hereinafter.

The formal notation used in the task of FIM for document relation discovery can be defined as follows. Let $\mathcal{D}$ be a set of documents (items) where $\mathcal{D} = \{d_1, d_2, ..., d_m\}$, and T be a set of terms (transactions) where $\mathcal{T} = \{t_1, t_2, ..., t_n\}$. Also, let $w(d_i, t_j)$ represent a weight of a term t_j in a document d_i. A subset of $\mathcal{D}$ is called a docset whereas a subset of $\mathcal{T}$ is called a termset. Furthermore, a docset $X_k = \{x_1, x_2, ..., x_k\} \subset \mathcal{D}$ with k documents is called k-docset.

Unlike most of FIM works on boolean-valued databases, this work also addresses the issue of mining frequent docsets from a real-valued database. In the task of mining frequent docsets, minimum support (a user-specified threshold) is used to filter out the docsets which have a support lower than this threshold, considered as infrequent docsets. Traditionally, the support of a docset is defined by a ratio between the number of terms that exist in all documents in the docset and the total number of distinct terms in a database. To this end, the support definition for a docset X_k is defined as follows.

$$sup(X_k) = \frac{\sum_{j=1}^{n} min_{i=1}^{k} w(x_i, t_j)}{\sum_{j=1}^{n} max_{i=1}^{m} w(d_i, t_j)} \tag{3.1}$$

By representing the data to be mined as in Figure 3.2, the new definition of support employs the *min* operation to find the weight of each term for a docset by selecting a minimum weight of the term among all documents in the docset. The *max* operation is applied for finding the maximum weight of each term in the database. The support of a docset will then be calculated from the ratio between the sum of all term weights for a docset and the sum of maximum weights of all terms in the database.

	d_1	d_2	d_3	d_4	$min\{w(d_2,t_j),w(d_3,t_j)\}$	$max_{i=1}^{4}\{w(d_i,t_j),w(d_i,t_j)\}$
t_1	1	1	0	0	0	1
t_2	1	1	1	1	1	1
t_3	1	1	1	1	1	1
t_4	1	1	0	1	0	1
sum	4	4	2	3	2	4

	d_1	d_2	d_3	d_4	$min\{w(d_2,t_j),w(d_3,t_j)\}$	$max_{i=1}^{4}\{w(d_i,t_j),w(d_i,t_j)\}$
t_1	4	2	0	0	0	4
t_2	4	2	4	1	2	4
t_3	2	4	2	2	2	4
t_4	2	4	0	1	0	4
sum	12	12	6	4	4	16

Figure 3.3: Example of support calculation on Boolean-valued (upper) and real-valued (lower) databases where upper: $sup(\{d_2,d_3\}) = \frac{2}{4}$ and lower: $sup(\{d_2,d_3\}) = \frac{4}{16}$

For clarity of explanation, let's consider the following example. According to the left of Figure 3.2, all terms in the database appear in d_2 while only term t_2 and t_3 appear in d_3. Assuming that we want to calculate the support of a docset d_2d_3 using the traditional support definition, the number of terms that contain d_2d_3 (i.e., 2) and the total number of terms (i.e., 4) need to be counted. The support of d_2d_3 is then equal to 2/4. This support value can also be calculated in another way as defined in the proposed generalized definition of support shown in the top of Figure 3.3. Employing the *min* operation on the weight of each term between column d_2 and d_3, the consequent weights for term t_1, t_2, t_3 and t_4 are 0, 1, 1 and 0, respectively. Applying the *max* operation on the weight of all terms among all documents, we will get one as the maximum weight for every term. Then, the support is the ratio between the sum of those minimum weights for d_2d_3 and the sum of maximum weight of all terms in the database, i.e., 2/4.

Although such Boolean-valued databases alone can be used to mine the docsets, there still is another kind of database which contains both term existence and term weighting, namely a real-valued database, as shown in the right of Figure 3.2. Unfortunately, the traditional support definition cannot be applied for calculating the support of a docset since it concerns only the term existence but not the term weight to a docset. To resolve this drawback, it is a good idea to use an equivalent definition of support as previously proposed. In this case, the support will represent the total minimum weight of co-occurring terms for all documents in a docset where each weight represents the importance such terms contribute to the docset. Each term weight for the docset with two or more documents will be equal to the minimum weight of the terms among all documents in a docset. This situation is conforming to our notion to differentiate the level of relations where the specified term weight for a docset must be lower than or equal to the weight of the terms for any documents in that docset. An example of support calculation for a docset d_2d_3 on real-valued database is illustrated in the bottom of Figure 3.3.

From the above statements, we can conclude that 1) the support value is still the same using either the traditional support definition or the generalized support definition for support calculating in a Boolean-valued database, and 2) the generalized support definition can also be used to calculate the support of any docset in a real-valued database. Using the databases

docset	generalized support	
	boolean-valued DB	real-valued DB
$\{d_1\}$	4/4	12/16
$\{d_2\}$	4/4	12/16
$\{d_3\}$	2/4	6/16
$\{d_4\}$	3/4	4/16
$\{d_1d_2\}$	4/4	8/16
$\{d_1d_3\}$	2/4	6/16
$\{d_1d_4\}$	3/4	4/16
$\{d_2d_3\}$	2/4	4/16
$\{d_2d_4\}$	3/4	4/16
$\{d_3d_4\}$	2/4	3/16
$\{d_1d_2d_3\}$	2/4	4/16
$\{d_1d_2d_4\}$	3/4	4/16
$\{d_2d_3d_4\}$	2/4	3/16
$\{d_1d_2d_3d_4\}$	2/4	3/16

Figure 3.4 Docsets and their supports (the boolean-valued v.s. the real-valued databases)

in Figure 3.2, the docsets and their supports, for boolean-valued and real-valued databases, can be computed as shown in Figure 3.4. Besides support, a so-called confidence is used for generating confident association rules. Here, the confidence is left since it is beyond the scope of this work.

Note that this generalized support preserves two closure properties as in [Agrawal et al., 1996], i.e., a downward closure property ("all subsets of a frequent itemset are also frequent"), and an upward closure property ("all supersets of an infrequent itemset are also infrequent"). For example, $sup(d_1) \geq sup(d_1d_2)$ and $sup(d_2) \geq sup(d_1d_2)$, if d_1d_2 is frequent then d_1 and d_2 are also frequent (downward closure property), and if either d_1 or d_2 or both of them are infrequent then d_1d_2 is also infrequent (upward closure property). The mathematical proof of closure property is given as follows.

Proof of closure property. Let X_{k-1} and X_k be the $\{k-1\}$-docset and k-docset, respectively, where $X_{k-1} \subseteq X_k$. In other words, $X_k = X_{k-1} \cup x_k$. It suffices to demonstrate that $sup(X_k) \leq sup(X_{k-1})$, i.e., the support of superset docset is less than the support of subset docset. Using the proposed definition of support, the support of x_{k-1} is:

$$sup(X_{k-1}) = \frac{\sum_{j=1}^{n} min(w(x_1,t_j), w(x_2,t_j), ..., w(x_{k-1},t_j))}{\sum_{j=1}^{n} max_{i=1}^{m} w(d_i,t_j)}, \qquad (3.2)$$

where n is the number of all terms and m is the number of all documents in the database. For a docset X_k, we get

$$sup(X_k) = \frac{\sum_{j=1}^{n} min(w(x_1,t_j), w(x_2,t_j), ..., w(x_{k-1},t_j), w(x_k,t_j))}{\sum_{j=1}^{n} max_{i=1}^{m} w(d_i,t_j)} \qquad (3.3)$$

As present in Equation 3.2 and 3.3, the numerator of the support fraction is changed according to which docset is calculated, but its denominator is the same for every docset. Therefore, only the numerator in the equations is taken into consideration. Comparing with the support

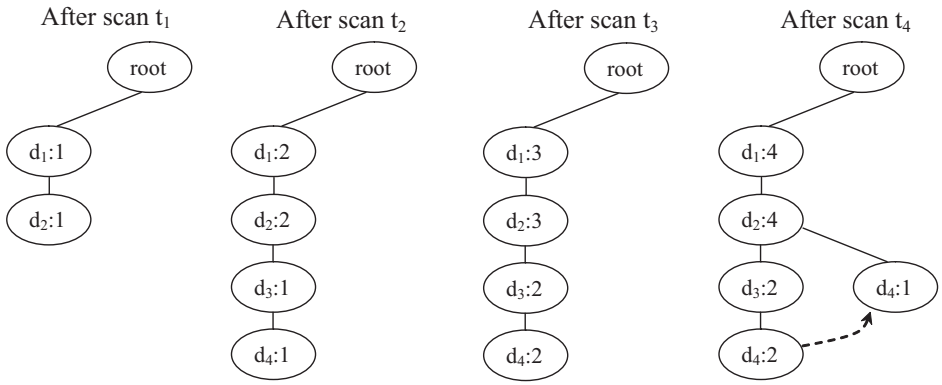


Figure 3.5 FP-Tree construction for Boolean-valued database in Figure 3.2

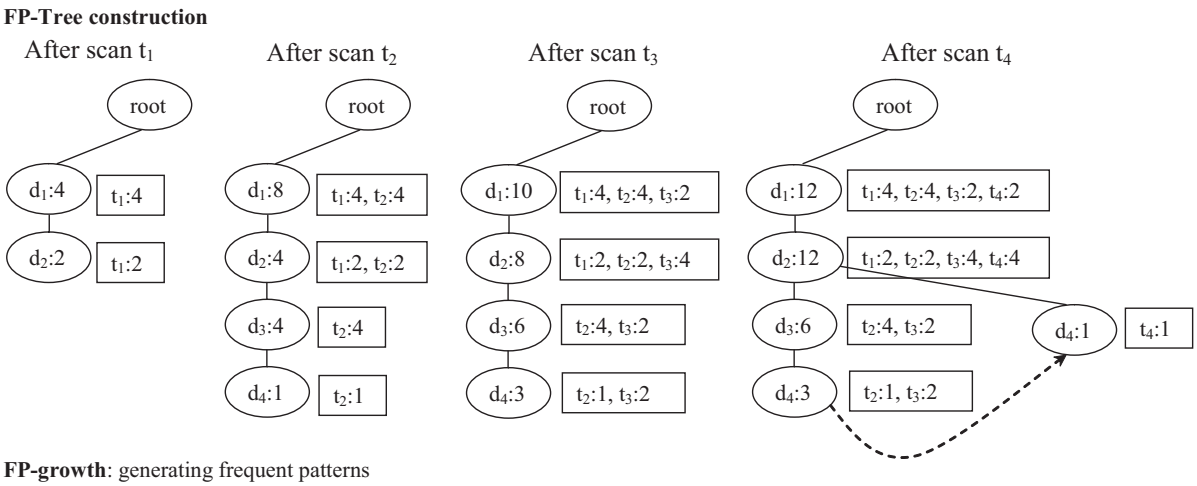
of X_{k-1}, one term $w(x_k, t_j)$ is added for the *min* operation in the support of X_k. With the property of *min* operation, $min(a,b) \leq min(a) and min(a,b) \leq min(b)$ given a, b as any real numbers. The weight is also a real number, therefore, the numerator of support X_k is less than the numerator of support X_{k-1}. Then, we get $sup(X_k) \leq sup(X_{k-1})$.

So far these properties have been applied in most existing FIM algorithms to reduce large computational time. Applying this modified frequent itemset mining is a promising approach for efficiently discovering all groups of document relations in a large collection of documents.

In this work, the FP-Tree algorithm [Han et al., 2000, Han et al., 2004] is used as a method for discovering the document relations. The algorithm is divided into two tasks, i.e. 1) FP-Tree construction and 2) FP-growth: mining frequent patterns with FP-Tree by pattern fragment growth, as described in Section 2.3.3. The Boolean-valued database can be mined by the original algorithm, but the real-valued database needs some modifications. Let's consider the example boolean-valued database in Figure 3.2. Using the original algorithm, the FP-Tree construction after scanning each transaction is shown in Figure 3.5. With the last constructed FP-Tree, the task of FP-growth will generate all frequent patterns straightforwardly. For a real-valued database as in Figure 3.2, the original FP-Tree algorithm needs to be modified as follows. In the task of FP-Tee construction, a set of real values in each transaction is stored with their corresponding items in the FP-Tree nodes when scanning the database. This extension makes FP-Tree to keep the information of real values that will be further used in the process of mining frequent patterns. The count of each node in the tree is accumulated in every transaction scanning. For the FP-growth task, the new definition of support, which is proposed in Equation 3.1, is employed to find all frequent patterns from the constructed FP-Tree. The generation is similar to the original algorithm but only their supports are calculated from the summation of *min* value between the real-valued of same transaction of the documents in the candidate patterns. The illustration of extended algorithm is presented in Figure 3.6.

3.2 Computational Time and Memory Usage

The computational time of the proposed algorithm to discover document relations is quite close to the computational time of the FP-Tree algorithm. As stated in [Han et al., 2004],



FP-growth: generating frequent patterns

$d_1 = 12$
$d_2 = 12$
$d_3 = 6$
$d_4 = 4$
$d_1d_2 = \min(4,2)+\min(4,2)+\min(2,4)+\min(2,4) = 8$
$d_1d_3 = \min(4,4)+\min(2,2) = 6$
$d_1d_4 = [1^{st}\text{ Path}]\ \min(4,1)+\min(2,2) + [2^{nd}\text{ Path}]\ \min(2,1) = 4$
$d_2d_3 = \min(2,4)+\min(4,2) = 4$
$d_2d_4 = [1^{st}\text{ Path}]\ \min(2,1)+\min(4,2) + [2^{nd}\text{ Path}]\ \min(4,1) = 4$
$d_3d_4 = \min(4,1)+\min(2,2) = 3$
$d_1d_2d_3 = \min(4,2,4)+\min(2,4,2) = 4$
$d_1d_2d_4 = [1^{st}\text{ Path}]\ \min(4,2,1)+\min(2,3,2)+[2^{nd}\text{ Path}]\ \min(2,4,1) = 4$
$d_2d_3d_4 = \min(2,4,1)+\min(4,2,2) = 3$
$d_1d_2d_3d_4 = \min(4,2,4,1)+\min(2,4,2,2) = 3$

Figure 3.6: Modified FP-Tree for real-valued database in Figure 3.2: FP-Tree construction (top) and FP-growth (bottom)

the computational cost for the FP-Tree algorithm is divided into two phases, i.e., the cost for constructing FP-Tree and the cost for mining frequent patterns. In the first phase, the computational cost of inserting transaction T into the FP-tree is $O(|freq(T)|)$, where $freq(T)$ is the set of frequent items in the transactions. The construction needs two scans of a database. In the latter phase, the modifications of FP-growth slightly affects the time complexity of an original algorithm. The frequent patterns can be generated using the constructed FP-Tree with the cost of $O(\sum_{i=1}^{n} \frac{|t_i|^2 \times \alpha}{2})$, where $|t_i|$ is the length of the transaction, α is the average number of real values attached to the FP-Tree nodes, and n is approximated as the number of transactions. If the number of transaction is large and the length of each transaction is long, the computation is quite costly. By focusing on the term-document database, the database is quite dense in comparison with the relational database. However, the average length of transactions is high since a term may appear in several documents. This phenomenon will affect the computational time of the second phase to generate the frequent patterns. Although the computational time of an algorithm is not a main study in this work, some investigations on the exact computational time is presented in Chapter 5.

However, there is another advantage of FP-Tree algorithm which is suitable for our approach. With its highly compact structure to store all information for frequent-patterns, the memory usage of the algorithm is less than or equal to the database size. Since there are often a lot of sharing of frequent items among transactions, the size of the FP-Tree is usually much smaller than its original database, and hence, small memory is used for handling such tree. This is very useful when we implement on a large-scaled document collection. Furthermore, it can be extended for adaptive learning without re-scanning the database when the new documents are added to the database.

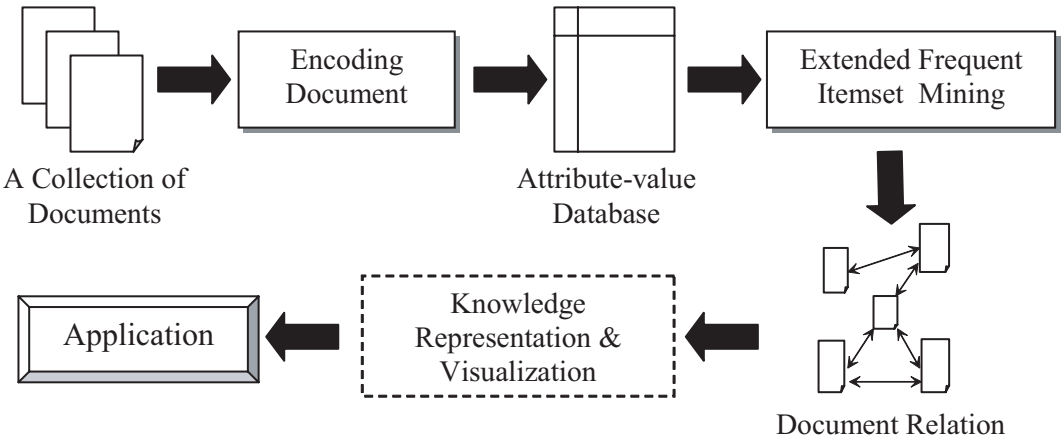


Figure 3.7 A framework of document relation discovery

3.3 Framework of Document Relation Discovery

In this thesis, a framework to discover the document relations is presented by utilizing the extended frequent itemset mining approach as shown in Figure 3.7. From the figure, there are three main processes for document relation discovery. First, a collection of documents will be encoded by several document representation models. This process produces an attribute-value database that can represent whole document contents in the collection. Second, the document relations can be discovered from such an encoded database using the extended frequent itemset mining. In the last process, the approaches of knowledge representation and visualization can be applied to utilize the discovered relations on the specific application, e.g., document relationship network or search engine. The thesis will focus on the first and the second processes while the last processes are left as the future work. However, the quality of discovered document relations is a mandatory issue to study since it is necessary to judge the performance of the model used for encoding the documents in the first process. The hypothesis of this thesis is that:

"By encoding a collection of documents as attribute-value database, the document relations can be discovered using the extended frequent itemset mining approach. The quality of discovered document relations depends on how to encode a collection of documents. There are two main factors for representing the documents in the database, i.e., term definition and term weighting. Moreover, each factor also contains various schemes, and the combination of those schemes can produce several types of document representation. The suitable combinations of term definition and term weighting schemes can increase the performance of discovered document relations."

By this hypothesis, the following problems arise:

1. What are the possible factors used in the document representation model?

2. What is the suitable document representation model used for encoding the documents to provide the high-quality document relations?

3. How to judge the quality of discovered document relations?

Toward resolving these problems, the thesis studies several schemes that can be used as document representation including the method to evaluate the quality of discovered documents. The document representations that are suitable for document relation discovery will be investigated in the experiments.

3.4 Document Representation

Most traditional works on text processing, including IR and TC, showed that a bag of individual words alone is not good enough for representing the content of a text [Feldman et al., 1998, Rajman and Besançon, 1998]. Several enhancements have been proposed to provide more suitable representation via term definition and term weighting. Let's consider the following example. Supposing that the three documents A, B and C are the scientific publications, the common terms that are usually found in the publications, e.g., "introduction", "related work", "proposed method", "conclusion", etc., may occur in those documents. Obviously, these terms are too general and domain-independent, i.e., they do not convey the specific contents to the documents. With the extended frequent itemset mining approach, ABC will probably have high support and becomes frequent pattern, but it can not be judged as a good document relation. Toward resolving this problem, the concept of document representation used for encoding the term-document database is investigated. With the suitable document representation, the good terms for well representing the document contents can be defined with term definition schemes, and the level of contribution for each term in the document can be set by the term weighting schemes.

In this section, three schemes of term definition, i.e., word-level n-gram, stemming, and stopword removal, and three schemes of term weighting are described. In the database point of view, the first three schemes are used for defining attributes while the last three schemes involve how to assign a value to those attributes in an attribute-value database.

3.4.1 Term Definition

The content of a document can be represented by a set of words inside that document. However, different sets of words provide different levels of how well they can represent all contents in a document. The term definition is an approach to select the appropriate sets of terms for representing the document contents. There are three main factors for term definition, i.e., n-gram, stemming and stopword removal. The details of each factor are described as follows.

1. **N-gram**
   For the first factor, several n-gram representations can be applied to define the terms in a document collection. It is well-known [Baeza-Yates and Ribeiro-Neto, 1999] that single words in a text, called unigram (1-gram), may not be good enough to represent semantics of the text due to an ambiguity of individual words. Therefore, a higher word-level n-gram can be applied. For the word-level n-gram, a term is defined by a set of any consecutive n words. In this work, only unigram and bigram representations are preliminarily taken into account. Therefore, we can classify the n-gram factor as follows.

(a) *unigram* defines a term as an individual word in the document contents.

(b) *bigram* defines a term from any consecutive two words in the document contents.

(c) *other n-gram*, there are the other *n*-gram representation where *n* is higher than two (higher than bigram). Although it is possible to apply a higher *n*-gram for defining the terms, the exponential growth of the number of terms may cause a problem for the mining process. Therefore, we have a trick for selecting only bigrams that contain no stopword.

For example, given a part of a text ".. data mining and artificial intelligence ..", a set of unigrams, say "data", "mining", "and", "artificial" and "intelligence" can be extracted. By bigram representation (2-gram), the following terms will be obtained i.e., "data mining", "mining and","and artificial" and "artificial intelligence". It is obvious that some bigrams contain stopwords and provide less meaning than the pure bigrams that do not have stopwords. Therefore in this work, only the bigrams without stopwords are selected as the term definition, i.e., "data mining" and "artificial intelligence" in this case.

2. **Stemming**

   (a) *non-applying stemming* leaves the term as it is originally defined without stemming.

   (b) *applying stemming* is an approach for reducing inflected (or sometimes derived) words to their stem, base or root form, generally a written word form. The Porter stemming algorithm [Porter, 1980] is used in this work since it is already embedded as a feature in the BOW [McCallum, 1996] text processing tool. The details of the algorithm are given in Appendix B.

3. **Stopword removal**

   (a) *non-applying stopword removal* will not filter any terms from the extracted terms.

   (b) *applying stopword removal* is an approach to filter out the words or terms which are contained in a set of English stoplists. The stoplist of SMART system [Rocchio, 1971] which contains 524 English common words are used in this work since it is already applied to the BOW toolkit [McCallum, 1996]. The complete list of stopwords is presented in Appendix B.

Combination of term definition schemes

As an objective of this work, we investigate the combination of the three schemes for term definition. By assumption, some schemes may affect the other schemes which results in different characteristics of discovered document relations. In this work, the combinations of the three term definition factors which will be investigated are shown in Table 3.1.

In the table, each term definition scheme is expressed as a triplet. The first item represents the *n*-gram representation, where 'U' stands for unigram and 'B denotes bigram. The second item states whether stemming is applied or not, where either 'X' or 'O' are used to express the non-applying and applying, respectively. In the last item, the applying of stopword removal is also expressed by either 'X' or 'O' in the encoding pattern.

Table 3.1: Term definition schemes and their encoding patterns expressed as triplets: {*n*-gram}, {stemming} and {stopword removal}.

Encoding Pattern	Term definition scheme		
	n-gram	stemming	stopword removal
UXX	unigram	non-applying	non-applying
UOX	unigram	applying	non-applying
UXO	unigram	non-applying	applying
UOO	unigram	applying	applying
BXX	bigram	non-applying	non-applying
BOX	bigram	applying	non-applying
BXO	bigram	non-applying	applying
BOO	bigram	applying	applying

3.4.2 Term Weighting

A document can be viewed as a vector in a vector space model [Salton et al., 1975]. In this representation, each element in the vector is equivalent to a unique term associated with its weight. The term weighting is applied to set a level of contribution of each term to the document. Each term weighting can be described by the combinations of three factors, i.e., term frequency, collection frequency and vector normalization. The details of each factor are shown below.

1. **Term frequency**
   For the first factor, term frequency, there are three alternative principal components as follows.

   (a) *binary term frequency: bf* is nothing more than 1 for term presence and 0 for term absence in a document. This factor was already implemented in the previous experiments in all cases of term definition schemes.

   (b) *occurrence term frequency: tf* is the number of occurrence of term in a document. In other words, this is usually called term frequency (*tf*).

   (c) *augmented normalized term frequency: antf* is defined by $0.5 + 0.5 \times tf/tf_{max}$ where tf_{max} is the maximum term frequency in a document. This compensates for relatively high term frequency in the case of long documents and normalizes term frequency to lie between 0.5 and 1.0.

2. **Collection frequency**
   In case a term frequently occurs in many documents, using *tf* alone for finding document relations may have little discriminative power. The collection frequency factor is then used for term discrimination. The best terms for identifying document contents are those which able to distinguish certain individual documents from the remainder of the collection. Two principal components for the collection frequency factor are shown as follows.

   (a) *no collection frequency* is defined by the multiplier of 1 to the other factors of term weighting.

(b) *inverse document frequency: (idf)* is defined as $log(\frac{N}{n_j})$, i.e., the log of the inverse of the fraction of documents in the whole set that contain term j, where N is the total number of documents and n_j is the number of documents in which a term j is assigned. The *idf* favors terms that occur in relatively few documents.

3. **Vector normalization**
The document length also affects the discovered relations since the longer documents are more likely to be relevant, as they are more likely to contain co-occurring terms. Therefore, the vector normalization factor is another important factor to represent a document. The document should be treated as equally important regardless of the document length. There are two alternative components as follows.

   (a) *no normalization* is defined by the multiplier of 1 to the other factors of term weighting.

   (b) *cosine normalization* is defined by the ratio of current term weight and a factor representing Euclidean vector length.

   (c) *maximum weight normalization* is defined by the ratio of current term weight and a maximum term weight.

   It is not necessary to equalize the unit length of document vectors. Indeed, the normalization can be applied to a whole term weighting or just a part of term weighting. For example, if the term weighting is $tf \times idf$, the normalization can be either tf_{rms} or $(tf \times idf)_{rms}$. This difference is interesting to study the outcomes when a large variety of vector normalization will be applied. Note that, the *rms* of a value stands for the root mean square of that value in a document which is coincident with cosine normalization. For example, given a document with m terms, tf_{rms} is defined by $\sqrt{\frac{\sum_{i=1}^{m} tf_i^2}{m}}$.

Combination of term weighting schemes

Several approaches on IR and TC [Jones, 1972, Salton and Yang, 1973, Wu and Salton, 1981, Yu et al., 1982, Salton and Buckley, 1988, Buckley, 1993, Zhang and Nguyen, 2005] succeeded in applying term weighting in several arithmetic forms. In those works, the term frequency and inverse document frequency are the prominent components used for setting a level of contribution and importance of a term to a document. However, there is an uncertainty of the combination on term weighting schemes that provide the best term weighting in identifying the document contents and are suitable for discovering document relations. Therefore, several combinations of term weighting schemes in Section 3.4.2 are studied in this chapter.

There are several combinations of the three term weighting factors, but some weighting schemes do not make sense to apply with the other schemes. For example, binary term frequency is meaningless to apply with cosine normalization (since the normalization is always one), or it is unnecessary to exploit the normalization to the augmented term frequency. To this end, the interesting term weighting schemes that will be explored in our experiments are selected as shown in Table 3.2.

In the table, each term weighting scheme is expressed as a triplet. The first item represents the usage of term frequency, where 'b' stands for binary term frequency, 't' denotes the occurrence term frequency and 'a' means augmented normalized term frequency. The second

Table 3.2: Term weighting schemes and their encoding patterns expressed as triplets: {term frequency}, {collection frequency} and {normalization}.

Encoding Pattern	Term weighting scheme			Equation
	term frequency	collection frequency	normalization	
bxx	*bf*	non-applying	non-applying	bf
bix	*bf*	*idf*	non-applying	$bf \times idf$
txx	*tf*	non-applying	non-applying	tf
tix	*tf*	*idf*	non-applying	$tf \times idf$
txc	*tf*	non-applying	*cosine*	$\frac{tf}{tf_{rms}}$
txm	*tf*	non-applying	*max tf*	$\frac{tf}{tf_{max}}$
tic	*tf*	*idf*	*cosine*	$\frac{tf}{tf_{rms}} \times idf$
tim	*tf*	*idf*	*max tf*	$\frac{tf}{tf_{max}} \times idf$
axx	*antf*	non-applying	non-applying	$0.5 + 0.5\frac{tf}{tf_{max}}$
aix	*antf*	*idf*	non-applying	$\left(0.5 + 0.5\frac{tf}{tf_{max}}\right) \times idf$

item shows whether the collection frequency is applied or not, where 'i' means *idf* is applied while 'x' signifies no collection frequency and multiply such term weighting scheme by 1. The last item indicates which normalization scheme is used, where 'x' is no normalization, 'c' represents cosine normalization and 'm' means maximum weight normalization. For example, 'aix' represents a term weighting scheme using augmented normalized term frequency, *idf* and no normalization.

3.5 Mining on Attribute-Value Database: Some Examples

Table 3.3 Example attribute-value database for illustrating document relation discovery

	d_1	d_2	d_3	d_4	d_5
t_1	5	4	2	0	0
t_2	0	2	0	0	0
t_3	3	2	3	0	0
t_4	1	0	0	6	0
t_5	2	0	1	0	0
t_6	4	5	0	8	3
t_7	4	0	0	0	0

For clarity, the approach of applying extended frequent itemset mining on attribute-value database, the sample calculations on a term-document database are illustrated in this section.

Given the database as shown in Table 3.3 , d_i is a document and t_j is a term that appears in a document where i and j are positive integers. Each nominal value in the table represents the term frequency of a term t_j in the specific document d_i called $w(d_i, t_j)$. Using this database, the calculations of document relations and their supports regarding to the concept in Section 3.1 can be applied on several term weighting schemes as follows.

1. Mining based on *binary term frequency, no collection frequency and no normalization (bxx)*: *bf*

Original database

	d_1	d_2	d_3	d_4
t_1	5	4	2	0
t_2	0	2	0	0
t_3	3	2	3	0
t_4	1	0	0	6
t_5	2	0	1	0
t_6	4	5	0	8
t_7	4	0	0	0

Database after applying term weighting scheme

	d_1	d_2	d_3	d_4	max
t_1	1	1	1	0	1
t_2	0	1	0	0	1
t_3	1	1	1	0	1
t_4	1	0	0	1	1
t_5	1	0	1	0	1
t_6	1	1	0	1	1
t_7	1	0	0	0	1
$\sum$	6	4	3	2	7
%support	85.71 ($\frac{6}{7} \times 100$)	57.14	42.86	28.57	

Document relations and their support calculations

	d_1d_2	d_1d_3	d_1d_4	d_2d_3	d_2d_4	d_3d_4	$d_1d_2d_3$	$d_1d_2d_4$	$d_1d_3d_4$	$d_2d_3d_4$	$d_1d_2d_3d_4$
t_1	1	1	0	1	0	0	1	0	0	0	0
t_2	0	0	0	0	0	0	0	0	0	0	0
t_3	1	1	0	1	0	0	1	0	0	0	0
t_4	0	0	1	0	0	0	0	0	0	0	0
t_5	0	1	0	0	0	0	0	0	0	0	0
t_6	1	0	1	0	1	0	0	1	0	0	0
t_7	0	0	0	0	0	0	0	0	0	0	0
$\sum$	3	3	2	2	1	0	2	1	0	0	0
%support	42.86	42.86	28.57	28.57	14.29	0.00	28.57	14.29	0.00	0.00	0.00

2. Mining based on *binary term frequency, idf and no normalization (bix)*: $bf \times idf$

Original database

	d_1	d_2	d_3	d_4	*idf*
t_1	5	4	2	0	0.1249 ($\log \frac{4}{3}$)
t_2	0	2	0	0	0.6021
t_3	3	2	3	0	0.1249
t_4	1	0	0	6	0.3010
t_5	2	0	1	0	0.3010
t_6	4	5	0	8	0.1249
t_7	4	0	0	0	0.6021

Database after applying term weighting scheme

	d_1	d_2	d_3	d_4	max
t_1	0.1249	0.1249	0.1249	0	0.1249
t_2	0	0.6021	0	0	0.6021
t_3	0.1249	0.1249	0.1249	0	0.1249
t_4	0.3010	0	0	0.3010	0.3010
t_5	0.3010	0	0.3010	0	0.3010
t_6	0.1249	0.1249	0	0.1249	0.1249
t_7	0.6021	0	0	0	0.6021
$\sum$	1.5786	0.9768	0.5508	0.4259	2.1807
%support	72.39	44.79	25.26	19.53	

Document relations and their support calculations

	d_1d_2	d_1d_3	d_1d_4	d_2d_3	d_2d_4	d_3d_4	$d_1d_2d_3$	$d_1d_2d_4$	$d_1d_3d_4$	$d_2d_3d_4$	$d_1d_2d_3d_4$
t_1	0.1249	0.1249	0	0.1249	0	0	0.1249	0	0	0	0
t_2	0	0	0	0	0	0	0	0	0	0	0
t_3	0.1249	0.1249	0	0.1249	0	0	0.1249	0	0	0	0
t_4	0	0	0.3010	0	0	0	0	0	0	0	0
t_5	0	0.3010	0	0	0	0	0	0	0	0	0
t_6	0.1249	0	0.1249	0	0.1249	0	0	0.1249	0	0	0
t_7	0	0	0	0	0	0	0	0	0	0	0
$\sum$	0.3747	0.5508	0.4259	0.2498	0.1249	0	0.2498	0.1249	0	0	0
%support	17.18	25.26	19.53	11.46	5.73	0.00	11.46	5.73	0.00	0.00	0.00

3. Mining based on *tf, idf and no normalization (tix)*: $tf \times idf$

Original database

	d_1	d_2	d_3	d_4	*idf*
t_1	5	4	2	0	0.1249
t_2	0	2	0	0	0.6021
t_3	3	2	3	0	0.1249
t_4	1	0	0	6	0.3010
t_5	2	0	1	0	0.3010
t_6	4	5	0	8	0.1249
t_7	4	0	0	0	0.6021

Database after applying term weighting scheme

	d_1	d_2	d_3	d_4	max
t_1	0.6245	0.4996	0.2498	0	0.6245
t_2	0	1.2042	0	0	1.2042
t_3	0.3747	0.2498	0.3747	0	0.3747
t_4	0.3010	0	0	1.8060	1.8060
t_5	0.6020	0	0.3010	0	0.6020
t_6	0.4996	0.6245	0	0.9992	0.9992
t_7	2.4084	0	0	0	2.4084
$\sum$	4.8102	2.5781	0.9255	2.8052	8.0190
%support	59.99	32.15	11.54	34.98	

Document relations and their support calculations

	d_1d_2	d_1d_3	d_1d_4	d_2d_3	d_2d_4	d_3d_4	$d_1d_2d_3$	$d_1d_2d_4$	$d_1d_3d_4$	$d_2d_3d_4$	$d_1d_2d_3d_4$
t_1	0.4996	0.2498	0	0.2498	0	0	0.2498	0	0	0	0
t_2	0	0	0	0	0	0	0	0	0	0	0
t_3	0.2498	0.3747	0	0.2498	0	0	0.2498	0	0	0	0
t_4	0	0	0.3010	0	0	0	0	0	0	0	0
t_5	0	0.3010	0	0	0	0	0	0	0	0	0
t_6	0.4996	0	0.4996	0	0.6245	0	0	0.4996	0	0	0
t_7	0	0	0	0	0	0	0	0	0	0	0
$\sum$	1.2490	0.9255	0.8006	0.4996	0.6245	0	0.4996	0.4996	0	0	0
%support	15.58	11.54	9.98	6.23	7.79	0	6.23	6.23	0	0	0

4. Mining based on *tf, idf and cosine normalization (tic)*: $\frac{tf}{tf_{rms}} \times idf$

Original database

	d_1	d_2	d_3	d_4	*idf*
t_1	5	4	2	0	0.1249
t_2	0	2	0	0	0.6021
t_3	3	2	3	0	0.1249
t_4	1	0	0	6	0.3010
t_5	2	0	1	0	0.3010
t_6	4	5	0	8	0.1249
t_7	4	0	0	0	0.6021
tf_{rms}	3.44	3.50	2.16	7.07	

Database after applying term weighting scheme

	d_1	d_2	d_3	d_4	max
t_1	0.1815	0.1427	0.1156	0	0.1815
t_2	0	0.3441	0	0	0.3441
t_3	0.1089	0.0714	0.1735	0	0.1735
t_4	0.0875	0	0	0.2554	0.2554
t_5	0.1750	0	0.1393	0	0.1750
t_6	0.1452	0.1784	0	0.1413	0.1784
t_7	0.7001	0	0	0	0.7001
$\sum$	1.3983	0.7366	0.4284	0.3967	2.0080
%support	69.64	36.68	21.34	19.76	

Note: *rms* is the root mean square of term weights for a specific document where the weight 0 is not taken into account.

Document relations and their support calculations

	d_1d_2	d_1d_3	d_1d_4	d_2d_3	d_2d_4	d_3d_4	$d_1d_2d_3$	$d_1d_2d_4$	$d_1d_3d_4$	$d_2d_3d_4$	$d_1d_2d_3d_4$
t_1	0.1427	0.1156	0	0.1156	0	0	0.1156	0	0	0	0
t_2	0	0	0	0	0	0	0	0	0	0	0
t_3	0.0714	0.1089	0	0.0714	0	0	0.0714	0	0	0	0
t_4	0	0	0.0875	0	0	0	0	0	0	0	0
t_5	0	0.1393	0	0	0	0	0	0	0	0	0
t_6	0.1452	0	0.1413	0	0.1413	0	0	0.1413	0	0	0
t_7	0	0	0	0	0	0	0	0	0	0	0
$\sum$	0.3593	0.3638	0.2288	0.1870	0.1413	0	0.1870	0.1413	0	0	0
%support	17.89	18.12	11.39	9.31	7.04	0	9.31	7.04	0	0	0

5. Mining based on *augmented normalized tf, idf and no normalization (aix)*: $\left(0.5+0.5\frac{tf}{tf_{max}}\right) \times idf$

Original database

	d_1	d_2	d_3	d_4	*idf*
t_1	5	4	2	0	0.1249
t_2	0	2	0	0	0.6021
t_3	3	2	3	0	0.1249
t_4	1	0	0	6	0.3010
t_5	2	0	1	0	0.3010
t_6	4	5	0	8	0.1249
t_7	4	0	0	0	0.6021
tf_{max}	5	5	3	8	

Database after applying term weighting scheme

	d_1	d_2	d_3	d_4	max
t_1	0.5625	0.5500	0.5416	0.5000	0.5625
t_2	0.5000	0.6204	0.5000	0.5000	0.6204
t_3	0.5375	0.5250	0.5625	0.5000	0.5625
t_4	0.5301	0.5000	0.5000	0.6129	0.6129
t_5	0.5602	0.5000	0.5502	0.5000	0.5602
t_6	0.5500	0.5625	0.5000	0.5625	0.5625
t_7	0.7408	0.5000	0.5000	0.5000	0.7408
$\sum$	3.9810	3.7578	3.6543	3.6753	4.2217
%support	94.30	89.01	86.56	87.06	

Document relations and their support calculations

	d_1d_2	d_1d_3	d_1d_4	d_2d_3	d_2d_4	d_3d_4	$d_1d_2d_3$	$d_1d_2d_4$	$d_1d_3d_4$	$d_2d_3d_4$	$d_1d_2d_3d_4$
t_1	0.5500	0.5416	0.5000	0.5416	0.5000	0.5000	0.5416	0.5000	0.5000	0.5000	0.5000
t_2	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000
t_3	0.5250	0.5375	0.5000	0.5250	0.5000	0.5000	0.5250	0.5000	0.5000	0.5000	0.5000
t_4	0.5000	0.5000	0.5301	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000
t_5	0.5000	0.5502	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000
t_6	0.5500	0.5000	0.5500	0.5000	0.5625	0.5000	0.5000	0.5500	0.5000	0.5000	0.5000
t_7	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000
$\sum$	3.6250	3.6293	3.5801	3.5666	3.5625	3.5000	3.5666	3.5500	3.5000	3.5000	3.5000
%support	85.87	85.97	84.80	84.48	84.39	82.91	84.48	84.09	82.91	82.91	82.91

Chapter 4

Automatic Evaluation of Document Relations

To evaluate the discovered document relations, this work presents an evaluation method to show the effectiveness of our technique. In the automatic evaluation, we use a citation graph, and evaluate our system based on its ability to find the relations that exist in the citation graph. In the manual evaluation, we confirm that some models of our approach can discover the relations which are highly coincident with human intuition.

Although human judgment is the best method for evaluation, it is a labor-intensive and time-consuming task. Therefore, the method for automatic evaluation using citation graph is a good alternative for evaluating the word-based model for document relation discovery. To accomplish the automatic evaluation method, firstly we define a citation graph and how to use it to evaluate, and secondly we discuss the evaluation by humans and the evaluation by the citation graph.

Intuitively, two documents are expected to be related under one of three basic situations: (1) one document cites to the other (direct citation), (2) both documents cite to the same document (bibliographic coupling) [Small, 1973] and (3) both documents are cited by the same document (co-citation) [Kessler, 1963]. An analysis of citation has been applied for several applications [White and McCain, 1989, Nanba et al., 2000, Rousseau and Zuccala, 2004].

Besides these basic situations, two documents may be related to each other via a more complicated concept called transitivity. For example, if a document A cites to a document B, and the document B cites to a document C, then one could assume a transitive relation between A and C. In this work, with the transitivity property, the concept of order citation is originally proposed to express an indirect connection between two documents. With the assumption that a direct or indirect connection between two documents implies a topical relation between them, such connection can be used for evaluating the results of document relation discovery.

In the rest of this chapter, introductions of the u-th order citation and v-th order accumulative citation matrix are given. Then, the validity is proposed as a measure for evaluating discovered docsets using information in the citation matrix. Finally, the expected validity is mathematically defined by exploiting the concept of generative probability and estimation. The approach of human evaluation will be discussed again in the next chapter.

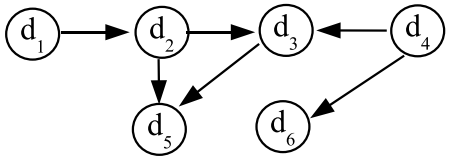


Figure 4.1 An example of a citation graph

4.1 The Citation Graph and Its Matrix Representation

Conceptually citations among documents in a scientific publication collection form a citation graph, where a node corresponds to a document and an arc corresponds to a direct citation of a document to another document. Based on this citation graph, an indirect citation can be defined using the concept of transitivity. The formulation of direct and indirect citations can be given in the terms of the u-th order citation and the v-th order accumulative citation matrix as follows.

Definition 1 *(**the u-th order citation**): Let $\mathcal{D}$ be a set of documents (items) in the database. For $x, y \in \mathcal{D}$, y is the u-th order citation of x iff the number of arcs in the shortest path between x to y in the citation graph is u (≥ 1). Conversely, x is also called the u-th order citation of y.*

For example, given a set of six documents $d_1, d_2, d_3, d_4, d_5, d_6 \in \mathcal{D}$ and a set of six citations, d_1 to d_2, d_2 to d_3 and d_5, d_3 to d_5, and d_4 to d_3 and d_6, the citation graph can be depicted in Figure 4.1. In the figure, d_1, d_3 and d_5 is the first, d_4 is the second, and d_6 is the third order citation of the document d_2. Note that although there is a direction for each citation, it is not taken into account since the task is to detect a document relation where the citation direction is not concerned. Moreover, using only textual information without explicit citation or temporal information, it is difficult to find the direction of the citation among any two documents.

Based on the concept of the u-th order citation, the v-th order accumulative citation matrix is introduced to express a set of citation relations stating whether any two documents can be transitively reached by the shortest path shorter than $v+1$.

Definition 2 *(**the v-th order accumulative citation matrix**): Given a set of n distinct documents, the v-th order accumulative citation matrix (for short, v-OACM) is an $n \times n$ matrix, each element of which represents the citation relation δ^v between two documents x, y where $\delta^v(x,y) = 1$ when x is the u-th order citation of y and $u \leq v$, otherwise $\delta^v(x,y) = 0$. Note that $\delta^v(x,y) = \delta^v(y,x)$ and $\delta^v(x,x) = 1$.*

For the previous example, the 1-, 2- and 3-OACMs can be created as shown in Figure 4.2. The 1-OACM can be straightforwardly constructed from the set of the first-order citation (direct citation). The $(v+1)$-OACM (mathematically denoted by a matrix A^{v+1}) can be recursively created from the operation between v-OACM (A^v) and 1-OACM (A^1) according to the following formula.

doc.	d_1	d_2	d_3	d_4	d_5	d_6
d_1	1	1	0	0	0	0
d_2	1	1	1	0	1	0
d_3	0	1	1	1	1	0
d_4	0	0	1	1	0	1
d_5	0	1	1	0	1	0
d_6	0	0	0	1	0	1

1-OACM

doc.	d_1	d_2	d_3	d_4	d_5	d_6
d_1	1	1	1	0	1	0
d_2	1	1	1	1	1	0
d_3	1	1	1	1	1	1
d_4	0	1	1	1	1	1
d_5	1	1	1	1	1	0
d_6	0	0	1	1	0	1

2-OACM

doc.	d_1	d_2	d_3	d_4	d_5	d_6
d_1	1	1	1	1	1	0
d_2	1	1	1	1	1	1
d_3	1	1	1	1	1	1
d_4	1	1	1	1	1	1
d_5	1	1	1	1	1	1
d_6	0	1	1	1	1	1

3-OACM

Figure 4.2 The 1-OACM (top), 2-OACM (middle) and 3-OACM (bottom)

$$a_{ij}^{v+1} = \vee_{k=1}^{n}(a_{ik}^{v} \wedge a_{kj}^{1}) \tag{4.1}$$

where $\vee$ is an OR operator, $\wedge$ is an AND operator, a_{ik}^{v} is the element at the i-th row and the k-th column of the matrix A^v and a_{kj}^{1} is the element at the k-th row and the j-th column of the matrix A^1. Note that any v-OACM is a symmetric matrix.

4.2 Validity: Quality of Document Relations

This section defines the validity which is used as a measure for evaluating the quality of the discovered docsets. The concept of validity calculation is to investigate how documents in a discovered docset are related to each other according to the citation graph. Based on this concept, the most preferable situation is that all documents in a docset directly cite to and/or are cited by at least one document in that docset, and thereafter they form one connected group. Since in practice only a few references are given in a document, it is quite rare and unrealistic that all related documents cite to each other. As a generalization, we can assume that all documents in a docset should cite to and/or are cited by each other within a specific range in the citation graph. Here, the shorter the specific range is, the more restrictive the evaluation is. With the concept of v-OACM stated in the previous section, we can realize this generalized evaluation by a so-called v-th order validity (for short, *v-validity*), where v corresponds to the specific range mentioned above.

Regarding the criteria of evaluation, two alternative scoring methods can be employed for

defining the validity of a docset. As the first method, a score is computed as the ratio of the number of citation relations in which the most popular document in a docset contains to its maximum. The most popular document is a document that has the most relations with the other documents in the docset. Note that, it is possible to have more than one most popular document in a docset. The score calculated by this method is called *soft validity*.

In the second method, a more strict criterion for scoring is applied. The score is set to 1 only when the most popular document connects to all documents in the docset. Otherwise, the score is set to 0. This score is called *hard validity*. The formulation of soft v-validity and hard v-validity of a docset X ($X \subset \mathcal{D}$), denoted by $S_S^v(X)$ and $S_H^v(X)$ respectively, are defined as follows.

$$\mathcal{S}_S^v(X) = \frac{max_{x \in X}(\sum_{y \in X, y \neq x} \delta^v(x,y))}{|X| - 1} \tag{4.2}$$

For simplicity, we denote a numerator in Equation 4.2 with $max^v(X)$, i.e., $max^v(X) = max_{x \in X}(\sum_{y \in X, y \neq x} \delta^v(x,y))$. Then,

$$\mathcal{S}_H^v(X) = \begin{cases} 1 & \text{, if } max^v(X) = |X| - 1 \\ 0 & \text{, otherwise} \end{cases} \tag{4.3}$$

Here, $\delta^v(x,y)$ is the citation relation defined by Definition 2 in Section 4.1. It can be observed that the soft v-validity of a docset is ranging from 0 to 1, i.e., $0 \leq \mathcal{S}_S^v(X) \leq 1$ while the hard v-validity is a binary value of 0 or 1. In both cases, the v-validity achieves the minimum (i.e., 0) when there is no citation relation among any document in the docset. On the other hand, it achieves the maximum (i.e., 1) when there is at least one document that has a citation relation with all documents in a docset. Intuitively, the validity of a bigger docset tends to be lower than a smaller docset since the probability that one document will cite to and/or be cited by other documents in the same docset becomes lower.

In practice, instead of an individual docset, the whole set of discovered docsets needs to be evaluated. The easiest method is to exploit an arithmetic mean. However, it is not fair to directly use the arithmetic mean since a bigger docset tends to have lower validity than a smaller one. We need an aggregation method that reflects docset size in the summation of validities. One of reasonable methods is to use the concept of weighted mean, where each weight reflects the docset size. Therefore, soft v-validity and hard v-validity for a set of discovered docsets $\mathcal{F}$, denoted by $\overline{\mathcal{S}_S}^v(\mathcal{F})$ and $\overline{\mathcal{S}_H}^v(\mathcal{F})$, respectively, can be defined as follows.

$$\overline{\mathcal{S}_S}^v(\mathcal{F}) = \frac{\sum_{X \in \mathcal{F}} w_X \times \mathcal{S}_S^v(X)}{\sum_{X \in \mathcal{F}} w_X} \tag{4.4}$$

$$\overline{\mathcal{S}_H}^v(\mathcal{F}) = \frac{\sum_{X \in \mathcal{F}} w_X \times \mathcal{S}_H^v(X)}{\sum_{X \in \mathcal{F}} w_X} \tag{4.5}$$

where w_X is the weight of a docset X. In this work, w_X is set to $|X| - 1$, the maximum value that the validity of a docset X can gain. For example, given the 1-OACM in Figure 4.2 and

$\mathcal{F} = \{d_1d_2, d_1d_2d_4\}$, the set soft 1-validity of $\mathcal{F}$ (i.e., $\overline{\mathcal{S}_S}^1(\mathcal{F})$) equals to $\frac{(1\times\frac{1}{1})+(2\times\frac{1}{2})}{1+2} = \frac{2}{3}$ while the set hard 1-validity of $\mathcal{F}$ (i.e., $\overline{\mathcal{S}_H}^1(\mathcal{F})$) is $\frac{(1\times\frac{1}{1})+(2\times 0)}{1+2} = \frac{1}{3}$.

The evaluation based on soft validity will focus on the probability that any two documents in a docset will occupy a valid relation. On the other hand, the evaluation based on hard validity will concentrate on the probability that at least one docset must have valid relations with all of the other documents. The soft validity states that how many valid relations in a set of discovered docsets based on the citation graph, while the hard validity identifies that how many perfect relations are there in a set of discovered relations based on citation graph. In this case, the perfect relation means the docset in which there is at least one document in the docset that contains valid relations with all of the other documents in that docset. Comparing between these two evaluation criteria, the hard validity is more restrictive than the soft validity, and the hard validity will always be lower than the soft validity in the same set of discovered docsets.

4.3 The Expected Validity

From Equations 4.2 and 4.3, the evaluation of discovered docsets will depend on the citation relation (δ^v), which is represented by v-OACMs. As stated in the previous section, the lower v is, the more restrictive the evaluation becomes. Therefore to compare the evaluation based on different v-OACMs, we need to declare a value, regardless of the restriction of evaluation, to represent the expected validity of a given set of docsets under each individual v-OACM. This section describes the method to estimate the theoretical validity of the set of docsets based on probability theory. Towards this estimation, the probability that two documents are related to each other under a v-OACM (later called *base probability*), needs to be calculated. This probability is derived by the ratio of the number of existing citation relations to the number of all possible citation relations in an OACM (i.e., $2\times\binom{|\mathcal{D}|}{2} = |\mathcal{D}|^2 - |\mathcal{D}|$) as shown in the following equation.

$$p_v = \frac{\sum_{x,y\in\mathcal{D},x\neq y}\delta^v(x,y)}{|\mathcal{D}|^2 - |\mathcal{D}|} \tag{4.6}$$

For example, using the citation relation in Figure 4.2, the base probabilities for 1-, 2-, and 3-OACMs are 0.40 (12/30), 0.73 (22/30) and 0.93 (28/30), respectively. Note that the base probability of a higher-OACM is always higher than or equal to that of a lower-OACM. Using the concept of expectation, the expected set v-validity ($E(\overline{\mathcal{S}}^v(\mathcal{F}))$) can be formulated as follows.

$$E(\overline{\mathcal{S}}^v(\mathcal{F})) = E(\frac{\sum_{X\in\mathcal{F}} w_X \times \mathcal{S}^v(X)}{\sum_{X\in\mathcal{F}} w_X}) \tag{4.7}$$

Since w_X and $\mathcal{S}^v(X)$ are independent, therefore

$$E(\overline{\mathcal{S}}^v(\mathcal{F})) = \frac{\sum_{X\in\mathcal{F}} E(w_X)\times E(\mathcal{S}^v(X))}{\sum_{X\in\mathcal{F}} E(w_X)} \tag{4.8}$$

Since w_X is a constant of $|X|-1$, the formula is then reduced to

$$E(\overline{\mathcal{S}}^{v}(\mathcal{F})) = \frac{\sum_{X \in \mathcal{F}} w_X \times E(\mathcal{S}^{v}(X))}{\sum_{X \in \mathcal{F}} w_X} \tag{4.9}$$

$$E(\mathcal{S}^{v}(X)) = \sum_{\forall Y_i, Y_i \in \beta(X)} (\mathcal{S}(Y_i) \times P^{v}(Y_i)) \tag{4.10}$$

where $E(\mathcal{S}^{v}(X))$ is the expected v-validity of a docset X, $\beta(X)$ is the set of all possible citation patterns for X, $S(Y_i)$ is the invariant validity of Y_i, and $P^{v}(Y_i)$ is the generative probability of the pattern Y_i estimated from the base probabilities under v-OACM (p_v). Theoretically, finding possible patterns of a docset can be transformed to the set enumeration problem. Given a docset with the length of k (k-docset), there are $2^{\binom{k}{2}}$ possible citation patterns.

With different scoring methods, an invariant validity is individually defined on each criteria regardless of the v-OACM. To simplify this, the notation $\mathcal{S}(Y_i)$ is replaced by $\mathcal{S}_S(Y_i)$ and $\mathcal{S}_H(Y_i)$ for the invariant validity calculated from soft validity and hard validity, respectively. Similar to Equation 4.2, an invariant validity of Y_i for soft validity is defined as follows:

$$\mathcal{S}_S(Y_i) = \frac{max_{x \in Y_i}(\sum_{y \in Y_i, y \neq x} \delta^{Y_i}(x,y))}{|Y_i| - 1} \tag{4.11}$$

For simplicity, we denote a numerator in the above equation by $max^{Y_i}(Y_i)$. With another case derived from Equation 4.3, an invariant validity of Y_i based on hard validity is given by:

$$\mathcal{S}_H(Y_i) = \begin{cases} 1 & \text{, if } max^{Y_i}(Y_i) = |Y_i| - 1 \\ 0 & \text{, otherwise} \end{cases} \tag{4.12}$$

In the above equations, $\delta^{Y_i}(x,y)$ is the citation relation between two documents x, y in the citation pattern Y_i where $\delta^{Y_i}(x,y) = 1$ when citation relation exists, otherwise $\delta^{Y_i}(x,y) = 0$. Note that all Y_i's have the same docset but represent different citation patterns. The following shows two examples of how to calculate the expected v-validity for 2-docsets and 3-docsets. For simplicity, the expected v-validity based on soft validity is firstly described, and the one based on hard validity is discussed later.

With the simplest case, there are only two possible citation patterns for a 2-docset. Therefore, the expected v-validity based on soft validity of any 2-docset (X) can be calculated as follows.

$$E(\mathcal{S}_S^{v}(X)) = \frac{1}{1} p_v + \frac{0}{1}(1 - p_v) = p_v \tag{4.13}$$

In the case of a 3-docset, there are eight possible patterns as shown in Figure 4.3. From Equation 4.11, we can calculate the invariant validity based on soft validity ($\mathcal{S}_S$) of each pattern as follows. The first to fourth patterns have the invariant validity of 1 (i.e., $\frac{2}{2}$). The fifth to seventh patterns gain the invariant validity of 0.5 (i.e., $\frac{1}{2}$) while the last pattern occupies the invariant validity of 0 (i.e., $\frac{0}{2}$). The generative probability of the first pattern is p_v^3 since there are three citation relations, and that of the second to the fourth patterns equals to $p_v^2(1 - p_v)$ since there are two citation relations and one missing citation relation. Regarding the citation pattern, the generative probabilities of the other patterns can be calculated in the

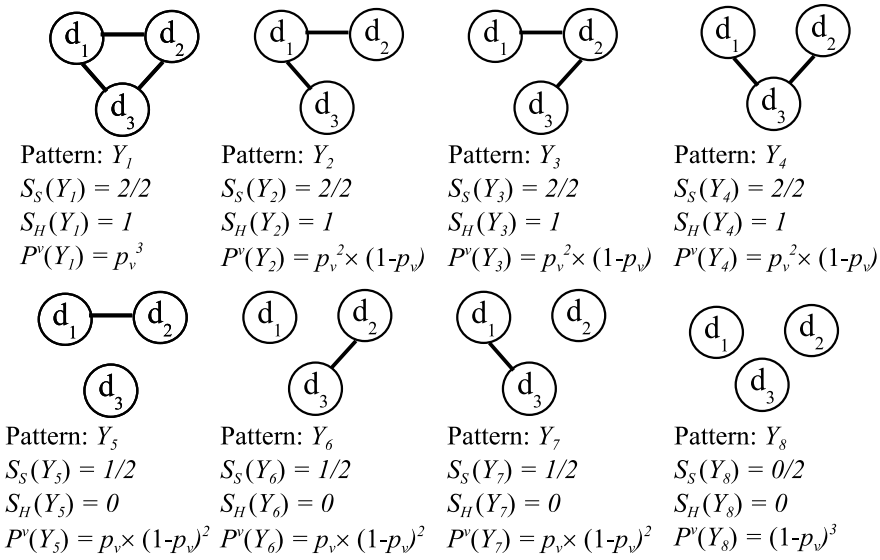


Figure 4.3 All possible citation patterns for a 3-docset

same manner. Applying Equation 4.10 and the generative probabilities shown in Figure 4.3, the expected *v*-validity based on soft validity can be calculated as follows.

$$E(\mathcal{S}_S^v(X)) = 1(\frac{2}{2}p_v^3) + 3(\frac{2}{2}p_v^2(1-p_v)) + 3(\frac{1}{2}p_v(1-p_v)^2) + 1(\frac{0}{2}(1-p_v)^3) \tag{4.14}$$

Here, the first term comes from the first pattern, the second term is derived from the second to the fourth patterns, the third term is obtained by the fifth to the seventh patterns and the last term is for the eighth pattern.

With another criterion of hard validity, the expected *v*-validity for a 2-docset is still the same but a difference occurs for a 3-docset. By Equation 4.12, the invariant validity based on hard validity ($\mathcal{S}_H$) equals to 1 for the first to fourth patterns and becomes 0 for the other patterns. The expected *v*-validity for a 3-docset based on hard validity is then reduced to

$$E(\mathcal{S}_H^v(X)) = 1(1 \times p_v^3) + 3(1 \times p_v^2(1-p_v)) \tag{4.15}$$

All above examples illustrate the calculation of the expected validity of only one docset. To calculate the expected *v*-validity of several docsets in a given set, the weighted mean of their validities can be derived by Equation 4.9. The outcome will be used as the expected value for evaluating the results obtained from our method for discovering document relations.

The pseudo-code for calculating the expected validity of a *k*-docset is given in Figure 4.4. This is the calculation for any docset with a specific *k* length. Given a specific *k* length, the algorithm starts from generating all binary relations, each of which links any two documents in a *k*-docset. In total, $\binom{k}{2}$ binary relations will be generated as a set A_k. All combinations of binary relations in A_k will be enumerated by a set enumeration and kept into β_k. For each element in β_k, the network from a set of binary relations can be formulated, and the maximum degree of node (document) will be selected as the *max* value. With a different scoring method, the invariant validity can be defined where the invariant soft validity is defined as

```
Input:        k (length of a docset to find its expected validity)
              p_v (base probability under v-OACM)
Output:       Expected validity of a k-docset
Method:    Call Cal_ExpVal(k, p_v).

Procedure Cal_ExpVal(k, p_v)
{
01      A_k = all binary relations for a k-docset: (k 2);
02      β_k = all enumerated combinations of binary relations in A_k (2^(k 2))
03      E=0;
04      For each element Y_i ∈ β_k    {
05        Formulating network from a set of binary relations Y_i;
06        max = maximum degree of node in a citation network;
07        if it is evaluated based on soft validity
08        then S(Y_i) = max;       //invariant soft validity
09        if it is evaluated based on hard validity
10        then if max = k − 1
11          then S(Y_i) = 1 else S(Y_i) = 0;   //invariant hard validity
12        n = ||Y_i||;        //number of binary relations in Y_i
13        P^v(Y_i) = p_v^n × (1 − p_v)^((k 2)−n);       //generative probability
14        E += S(Y_i) × P^v(Y_i);   }
15      output E;       //expected validity of a k-docset
}
```

Figure 4.4 Pseudo-code for calculating the expected validity of a k-docset under v-OACM

max value, but the invariant hard validity is defined to be 1 when *max* value is equal to the number of documents in a docset minus one, otherwise it becomes 0. Next, the generative probability is estimated by the multiplication of citation probabilities in the network based on the given p_v, base probability under v-OACM. Finally, the expected validity of a docset is then calculated by the summation of the multiplications between invariant validity and generative probability of each citation pattern in β_k.

Chapter 5

Experimental Results and Evaluations

This chapter presents a set of experimental results when the quality of discovered docsets is investigated under several empirical evaluation criteria. The four main objectives are (1) to investigate characteristics of the evaluation by soft validity and hard validity on docsets discovered from different document representations including their minimum support thresholds and mining time, (2) to study the quality of discovered relations when using either direct citation or indirect citation as the evaluation criteria, (3) to present the relative quality of a discovered relation when it is compared to its statistical expected value, and (4) to show the quality of discovered relations evaluated by humans and compare the results with those from the proposed evaluation method.

Towards the first objective, several term definitions are explored in the process of encoding the documents. To define terms in a document, techniques of n-gram, stemming and stopword removal can be applied. In this experiment, binary term weighting is the preliminary focus. The discovered docsets are ranked by their supports, and then the top-N ranked relations are evaluated using both soft validity and hard validity. Here, the value of N can be varied to observe the characteristic of the discovered docsets. For the second objective, the evaluation is performed based on various v-OACMs, where the 1-OACM considers only direct citation while a higher-OACM also includes indirect citation as shown in Chapter 4. Intuitively, the evaluation becomes less restricted when a higher-OACM is applied as the calibration. To fulfill the third objective, the expected set validity for each set of discovered relations is calculated according to the method shown in Section 4.3. Compared to this expected validity, the significance of discovered docsets is investigated. In the last objective, a set of discovered relations is sampled and evaluated by letting a number of experts rate the relatedness of documents in each relation. The result can be used to confirm the potential of our proposed evaluation method.

5.1 Experimental Setting

In this section, the construction of evaluation material and the characteristics of dataset are first described. After that, the preprocessing step for extracting terms from a document collection including the environments of the experiments are presented.

5.1.1 Evaluation Material

There is no gold standard dataset that can be used for evaluating the results of document relation discovery. To solve this problem, an evaluation material using a citation graph is formulated during the process of constructing the dataset. The dataset used in this work is a collection of scientific research publications retrieved from the ACM Digital Library. The

scientific publications and their citation information are iteratively collected as test data. By encoding the scientific publications as an attribute-value database, the document relations between the publications can be discovered by the extended frequent itemset mining as described in this work. During the process, the collected citation information can be formulated as the evaluation criteria for assessing the quality of discovered document relations. By this simple construction approach, it can be implemented on any collection of documents for discovering document relations and validating the quality of those relations. In this part, the detail of our dataset construction is described and the characteristics of the dataset used in the experiments, including the examples of publications and their citation information, are presented. Note that although the construction method is specific to this dataset, it is simple to apply it to other document collections.

Dataset Construction

Following the CCS (Computer Classification System[1]), three classes in computer related fields: Class B:Hardware, Class E:Data and Class J:Computer, are selected as three search keywords. Using the search engine of the ACM Digital Library[2] and giving the class as a query, the relevant documents can be retrieved, ordered by their relevancies to the class. For each of the three classes, the top 200 publications are collected as a seed in the dataset. In total, there are 600 publications at the beginning. For each publication, the PDF file format and its information page which identifies the citation (or reference) information are gathered. The reference publications appearing in those 600 publications are further collected and added into the dataset. In the same way, the publications referred to by these newly collected publications are also gathered and appended into the dataset. With three iterations, there are totally 10,817 publications used as a test collection. Some examples of publications and their references can be found in Appendix C.

With the use of the information page attached to each publication, the citation graph can be constructed and used for evaluating the discovered docsets. To control the characteristics of the citation graph which will influence the value of expected validity, only the citations between two publications where both of them present in our dataset are included in the citation graph. The 1-OACM can be encoded from this citation graph and then used to construct the 2-OACM and 3-OACM.

After converting these collected publications to ASCII text format, the reference (normally found at the end of each publication text) is removed by a semi-automatic process, such as using clue words of “References” and “Bibliography”. This operation makes the approach of document relation discovery fair to retrieve the high-quality relations with blind citation information. This dataset is now ready to use in any word-based approach for document relation discovery.

Characteristic of Dataset

There are two aspects of the dataset characteristics, i.e., term definition and citation. The characteristic in the aspect of term definition presents the number of terms in the datasets when several term definition schemes are applied. The characteristic in the aspect of citation shows the number of citation relations which are present in several OACMs that were constructed based on citation information of documents in the dataset.

[1] http://www.acm.org/class/

[2] http://www.portal.acm.org

Table 5.1 The number of terms in the dataset for each term definition patterns

Encoding Pattern	Term Definition			Number of all terms	Number of distinct terms with $tf > 1$	Number of distinct terms with $tf > 2$ (*AVG)
	Unigram/ Bigram	Stemming	Stopword removal			
UXX	Unigram	X	X	466,424	110,132	71,394 (43.00)
UXO	Unigram	X	O	463,664	108,161	69,686 (27.95)
UOX	Unigram	O	X	397,825	87,850	55,360 (52.40)
UOO	Unigram	O	O	395,630	86,234	53,941 (33.83)
BXX	Bigram	X	X	7,151,014	1,133,020	588,131 (4.69)
BXO	Bigram	X	O	3,866,543	544,315	283,673 (1.90)
BOX	Bigram	O	X	5,352,994	953,786	508,797 (5.54)
BOO	Bigram	O	O	2,875,442	472,209	253,440 (2.23)

*AVG is the average number of documents per term (number of items per transaction)

Table 5.2 The number of citation relations of the dataset in each OACM

	1-OACM	2-OACM	3-OACM
Total number of citation relations	83,945	1,600,507	11,020,918
Average number of citation relations per document	8	148	1,020
Maximum number of citation relations for a document	239	1,155	6,102
Minimum number of citation relations for a document	2	2	2

- *Characteristic in the Aspect of Term Definition*

  The value 'X' in the table means non-applying such term weighting schemes while the value 'O' means applying. The number of distinct terms with $tf > 1$ indicates the number of distinct terms where those terms have *tf* greater than 1, and the number of distinct terms with $tf > 2$ means the number of distinct terms where those terms have *tf* greater than 2. This illustration shows that the number of terms is dramatically reduced when we filter out the terms with low *tf*. Note that in our experiments, only the terms with $tf > 2$ are used as the document representation due to the limitation of the mining engine and the overwhelming number of insignificant terms. As shown in Table 5.1, the average length of transactions for bigram cases is approximately 10 times lower than those for unigram cases. This phenomenon is affected from our process in selecting only bigrams that contain no stopwords as stated in Section 3.4.1. The number of bigrams is dramatically reduce from the exponential number to the linear number when compared with unigrams. As a result, the computational time of mining in the bigram cases will not grow exponentially when the larger number of document relations is considered.

- *Characteristic in the Aspect of Citation*

  Note that the number of citation relations are counted based on the existing relations where both citer (cite from) and citee (cite to) documents are contained as their nodes and existed in the dataset.

5.1.2 Preprocessing Step

Together with text preprocessing, the BOW library [McCallum, 1996] is used as a tool for constructing a document-term database. Using a list of 524 stopwords [Salton and McGill, 1986], common words, such as *a*, *an*, *is* and *for*, are discarded. Besides these stopwords, terms with very low frequency are also omitted. These terms are numerous and usually negligible. Moreover, a term occurring less than three times is considered to be insignificant and thus pruned. By this process, the number of terms is dramatically reduced by a factor of 7 to 13. For instance, in the case of applying non-stemming, stopword removal and bigram, the number of terms reduces from 3,866,543 to 283,673 terms. From our observation, the remaining terms in a document still preserve the content of the document. In the case of using bigrams as terms, all bigrams are first generated from the original text, and then the bigrams which contain stopwords or have low frequency are pruned. This process will help us to generate pairs of consecutive words, e.g., compound nouns, without the insertion of stopwords.

5.1.3 Environments

To implement a mining engine for document relation discovery, the FP-tree algorithm, originally introduced in [Han et al., 2000], is modified to mine docsets in both binary-valued and real-valued databases as described in Chapter 3. In this work, instead of association rules, frequent itemsets are considered. Since a 1-docset contains no relation, only the discovered docsets with at least two documents are taken into account. The experiments were performed on a Pentium IV 2.4GHz Hyper-Threading with 1GB physical memory and 2GB virtual memory running Linux TLE 5.0 as an operating system. The preprocessing steps, i.e., n-gram construction, stemming and stopword removal, consume negligible computational time.

5.2 Results from Automatic Evaluation

As stated at the beginning of this chapter, several term definitions can be used as factors to obtain various patterns of document representation. In our experiment, eight distinct patterns of term definitions are explored. Each pattern is expressed as a triplet. The first item represents the usage of *n*-gram, where 'U' stands for unigram and 'B' means bigram. The second item has a value of either 'O' or 'X', expressing whether the stemming scheme is applied or not. Also the last item is either 'O' or 'X', telling us whether the stopword removal scheme is applied or not. For example, 'UXO' means document representation generated by unigram, non-stemming and stopword removal. In this chapter, the binary term frequency is mainly focused on as the term weighting scheme.

5.2.1 Evaluation based on 1-OACM

Table 5.3 expresses the set 1-validity (soft validity/hard validity) of the discovered docsets when various document representations are applied. The minimum support and the execution time of mining for each document representation to discover a specified number of top-*N* ranked docsets are also given in the table. From the table, some interesting observations can

Table 5.3: Set 1-validity for various top-N rankings of discovered docsets, their supports and mining time: soft validity/hard validity (upper: bigram, lower: unigram), MINSUP: MINIMUM SUPPORT ($\times 10^{-2}$) TIME: MINING TIME (SECONDS)

N	Set Validity (%)			
	BXO	BOO	BXX	BOX
1000	45.47/43.95	46.14/44.33	6.29/6.29	7.09/7.09
	MINSUP=0.53,TIME=174.49	MINSUP=0.67,TIME=155.92	MINSUP=3.94,TIME=442.95	MINSUP=4.76,TIME=402.14
5000	29.31/23.88	29.13/27.24	3.83/3.33	3.88/3.59
	MINSUP=0.35,TIME=188.88	MINSUP=0.47,TIME=166.96	MINSUP=3.15,TIME=612.82	MINSUP=3.79,TIME=570.65
10000	24.49/19.33	24.40/20.50	3.13/2.33	3.20/2.63
	MINSUP=0.32,TIME=189.52	MINSUP=0.39,TIME=170.17	MINSUP=2.84,TIME=681.40	MINSUP=3.42,TIME=627.61
50000	19.29/ 6.36	18.88/ 8.62	2.46/0.98	2.36/1.19
	MINSUP=0.25,TIME=195.39	MINSUP=0.29,TIME=176.48	MINSUP=2.31,TIME=816.43	MINSUP=2.71,TIME=767.25
100000	19.51/ 3.67	18.40/ 4.11	2.30/0.63	2.18/0.77
	MINSUP=0.21,TIME=212.14	MINSUP=0.28,TIME=176.57	MINSUP=2.13,TIME=862.84	MINSUP=2.48,TIME=832.77
Average	27.61/19.64	27.39/20.96	3.60/2.71	3.74/3.05
	MINSUP=0.33,TIME=192.08	MINSUP=0.42,TIME=169.22	MINSUP=2.87,TIME=683.29	MINSUP=3.43,TIME=640.08

N	Set Validity (%)			
	UXO	UOO	UXX	UOX
1000	3.88/3.78	2.36/2.26	2.79/2.79	1.76/1.76
	MINSUP=32.72,TIME=122.49	MINSUP=46.35,TIME=74.77	MINSUP=55.61,TIME=160.98	MINSUP=74.78,TIME=89.39
5000	3.77/3.35	2.38/1.99	2.37/2.28	1.55/1.48
	MINSUP=26.98,TIME=240.57	MINSUP=40.04,TIME=175.72	MINSUP=48.46,TIME=359.18	MINSUP=66.84,TIME=198.16
10000	3.47/2.63	2.16/1.53	2.09/1.75	1.35/1.11
	MINSUP=24.68,TIME=312.69	MINSUP=37.63,TIME=231.41	MINSUP=45.66,TIME=466.00	MINSUP=63.76,TIME=277.67
50000	2.78/1.44	1.75/0.74	1.68/0.84	1.12/0.49
	MINSUP=19.95,TIME=478.97	MINSUP=32.26,TIME=412.79	MINSUP=39.64,TIME=808.61	MINSUP=57.08,TIME=539.55
100000	2.71/1.02	1.68/0.48	1.66/0.57	1.14/0.32
	MINSUP=18.37,TIME=564.65	MINSUP=30.40,TIME=531.10	MINSUP=37.40,TIME=1008.38	MINSUP=54.55,TIME=691.02
Average	3.32/2.44	2.06/1.40	2.12/1.64	1.38/1.03
	MINSUP=24.54,TIME=343.87	MINSUP=37.34,TIME=285.16	MINSUP=45.35,TIME=560.63	MINSUP=63.40,TIME=359.16

be made. First, with the same document representation, soft validity is always higher than or equal to hard validity since the former is obtained by less restrictive evaluation than the latter (see Equation 4.2 and 4.3). Both validities involve valid relations between any pair of documents in a discovered docset. A relation between two documents is called valid when there is a link between those two documents under the v-OACM (v=1 in this experiment). The evaluation based on soft validity focuses on the probability that any two documents in a docset will occupy a valid relation. On the other hand, the evaluation based on hard validity concentrates on the probability that at least one docset must have valid relations with all of the other documents. For example, in the case of top-100000 ranking with the 'BXO' representation (as shown in Table 1), 19.51% of the relations in the discovered docsets are valid while only 3.67% of the discovered docsets are perfect, i.e., there is at least one document that contains valid relations with all of the other documents in the certain docset. Second, in every document representation, both soft validity and hard validity become lower when more ranks (i.e., top-N ranking with a larger N) are considered. As an implication of this result, our proposed evaluation method indicates that better docsets are located at higher ranks. Third, given two representations, say A and B, if the soft validity of A is better than that of B, then the hard validity of A tends to be higher than that of B. Fourth, the results

Table 5.4: The set 1-validity for each docset length when the top-100000 ranking is considered. Each cell indicates soft validity/hard validity, as well as the number of docsets (in the bracket)

Docset length	BXO	BOO	UXO	UXX
2	10.86/10.86 (40,870)	11.21/11.21 (38,553)	1.83/1.83 (64,326)	1.31/1.31 (55,262)
3	14.00/4.54 (30,679)	17.35/6.01 (26,174)	3.32/0.35 (33,489)	1.97/0.15 (40,934)
4	20.73/2.05 (10,759)	19.20/1.98 (18,593)	5.09/0.00 (2,181)	1.07/0.00 (3,798)
5	24.40/0.62 (8,004)	21.59/0.66 (13,084)	6.25/0.00 (4)	0.00/0.00 (6)
6	27.07/0.17 (5,266)	24.61/0.09 (3,519)		
7	28.83/0.04 (2,835)	41.31/0.00 (71)		
8	30.60/0.00 (1,168)	45.24/0.00 (6)		
9	32.67/0.00 (347)			
10	35.19/0.00 (66)			
11	38.33/0.00 (6)			
%Set validity	19.51/3.67	18.40/4.11	2.71/1.02	1.66/0.57

of the bigram cases ('B**') are much better than those of the unigram cases ('U**'). One reason is that the bigrams are quite superior to the unigrams in representing the content of a document. Fifth, in the cases of bigram, the stopword removal process is helpful while the stemming process does not help much. Sixth, in the cases of unigram, non-stemming is preferable while the stopword removal process is not useful. Finally, the performance of 'BXO' and 'BOO' is comparable and much higher than 'BOX' and 'BXX', while the performance of 'UXO' is much higher than the other unigram cases. However, on average, the 'UXX' seems to be the second best case for the unigram. Since the soft validity is more flexible than the hard validity, a higher soft validity is preferable. Although performance of 'BOO' seems to be slightly better than 'BXO' in the higher ranks, 'BXO' performs better on average. In our task, the performance for bigram is 'BXO' $>$ 'BOO' and the performance for unigram is 'UXO' $>$ 'UXX'.

In terms of minimum support and computation time, we can conclude as follows. First, since a docset discovered from the bigram cases tends to have a lower support than the unigram cases, it is necessary to set a small minimum support in order to obtain the same number of docsets. Second, the cases with stopword removal run faster than ones without stopword removal since they consider fewer words. Moreover, they tend to have a lower minimum support.

As a more detailed exploration of these four best cases, the soft validity and the hard validity as well as the number of discovered docsets for each docset length are investigated. The result of the top-100000 ranking is shown in Table 5.4. Due to the space limitation, the results of the other top-N rankings are omitted but they perform in similarly characteristics. From the table, some interesting characteristics are observed: (1) the number of bigger docsets is smaller, (2) compared to the unigram, the bigram produces bigger docsets, (3) in most

cases, the soft validity of bigger docsets is higher than that of smaller ones, while the hard validity of bigger docsets is lower than that of smaller ones. These observations reflect a good characteristic of the evaluation and match with our expectation.

5.2.2 Evaluation based on 1-, 2- and 3-OACMs

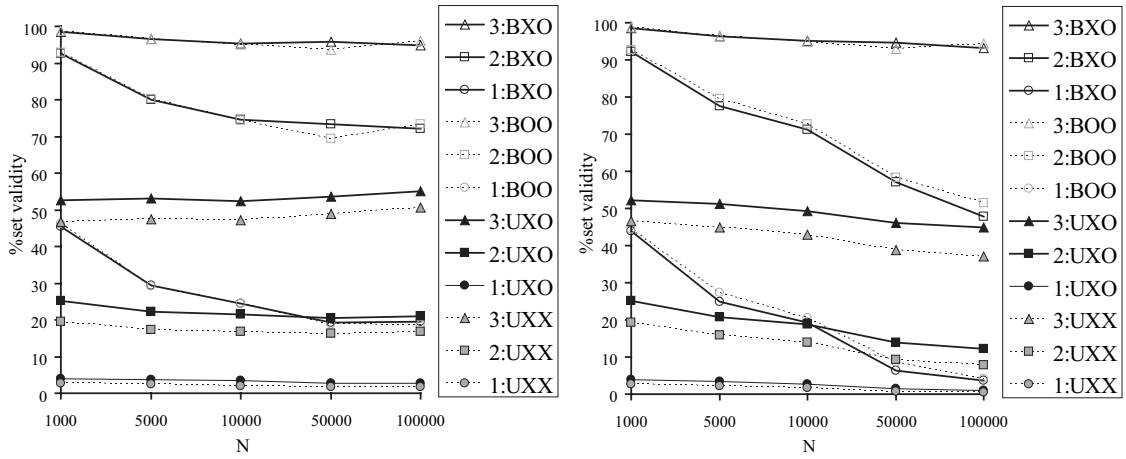


Figure 5.1: Set validity based on the 1-, 2- and 3-OACMs when various top-N rankings of discovered docsets are considered: soft validity (left) and hard validity (right)

Besides 1-OACM, the discovered docsets can be evaluated with the criteria of 2-OACM and 3-OACM. In this assessment, only the four best representations, two from the unigram cases ('UXO' and 'UXX') and two from the bigram cases ('BXO' and 'BOO'), are taken into consideration. Figure 5.1 displays the soft validity (the left graph) and the hard validity (the right graph) under 1-, 2-, and 3-OACMs. Since the minimum support and mining time in each case are the same as shown in Table 5.3, they are omitted from the figure. In the figure, we use the notation to represent the evaluation of docsets under the specified OACM where those docsets are discovered from a specific document representation. For example, '3:BXO' means the evaluation of docsets under 3-OACM where the docsets are discovered by encoding document representation using the BXO scheme (bigram, non-stemming and stopword removal). Being consistent for both soft validity and hard validity, the set 3-validity (one calculated under the 3-OACM) of discovered docsets is higher than the set 2-validity (one calculated under the 2-OACM), and in the same way the set 2-validity is much higher than the set 1-validity (one calculated under the 1-OACM). Compared to the evaluation using only direct citation (1-OACM), more relations in the discovered docsets are valid when both direct and indirect citations (2- and 3-OACMs) are taken into consideration.

Similar to 1-OACM, 'BXO' and 'BOO' are comparable and perform as the best cases for both soft validity and hard validity under the same OACM. Moreover, in the cases of bigram evaluated under the 1- and 2-OACMs, the set validity drops remarkably when the top-N rankings with a larger N are focused upon. The quality of docsets in the higher rank (smaller N) outperforms that of the lower rank. This outcome implies that our evaluation based on direct/indirect citations seems to be a reasonable method for assessing docsets. For all types of document representation, the bigram cases perform better than the unigram cases when they are evaluated under the same v-OACM. Especially the cases under 3-OACM, where

both two bigram cases ('3:BXO' and '3:BOO') are almost 100% valid while two unigram cases ('3:UXO' and '3:UXX') are approximately 50% valid. This phenomenon shows the advantage of bigram in being a good document representation for discovering document relations where the documents in each relation are likely to cite other documents under the specific range within citation graph. Furthermore, the performance gap between bigram and unigram becomes smaller when top-N rankings with a larger N are considered. For a top-N ranking with a larger N, the bigram cases tend to have bigger docsets than the unigram cases and then obtain lower validity since naturally a bigger docset is likely to have lower validity.

5.2.3 Actual Validity vs. Expected Validity

In the next experiment, the evaluation is made to investigate the relative quality of discovered docsets against the expected validity. As stated in Section 4.3, to compare the evaluation based on different v-OACMs, the expected validity can be calculated for each individual v-OACM. To do this, the expected set validity is calculated with respect to Equation 4.9. Using Equation 4.6, the base probabilities under 1-, 2-, and 3-OACMs (p_1, p_2 and p_3) for our collection are 6.26×10^{-4}, 1.36×10^{-2} and 9.41×10^{-2}, respectively. Due to the space limitation, only the investigations of 'BXO' and 'UXO' are shown here, but the other cases are similar to these two cases. The actual set validity gained from the experiments, the expected set validity calculated from Equation 4.9 and their ratio are displayed in Table 5.5 and Table 5.6, for soft validity and hard validity, respectively. The ratio expresses the quality of the discovered docsets compared to its expected validity.

From Tables 5.5 and 5.6, the quality of discovered docsets is significantly high, compared to the expected validity. In principle, the expected validity of a lower-OACM is always lower than or equal to that of a higher-OACM. For our collection, the expected validity of 2-OACM is approximately 20-22 times higher than that of 1-OACM, while the expected validity of 3-OACM is about 7-9 times higher than that of 2-OACM. Incidentally, this figure is obtained for both soft validity and hard validity. Although it seems that we gain a low set validity for a lower-OACM, if we compare that validity to its expected validity, we will find out that the ratio is considerably large. That is, the discovered docsets are eligible. For instance, focusing on the top-1000 ranking, although we gained approximately 4% for both soft validity and hard validity under the 1-OACM with the unigram ('UXO'), it corresponds to 60 times over the expected validity. Under the same condition, for the 2- and 3-OACM, we obtained approximately 19 and 6 times over the expected validity, respectively. In the case of bigram ('BXO') and under the 1-, 2- and 3-OACMs, the ratios are approximately 676, 65 and 10, respectively, for soft validity, while they raise to approximately 754, 74 and 11, respectively, for hard validity.

By comparing the result to the expected validity, the evaluations under different v-OACMs become comparable with fair evaluation. Although the set validity of discovered docsets under a lower-OACM is low, it may be relatively high compared to the expected validity. In Table 5.5 and 5.6, although the order of the set validities for different OACMs is 3-OACM $>$ 2-OACM $>$ 1-OACM for given discovered docsets, the order of their ratios is 1-OACM $>$ 2-OACM $>$ 3-OACM. This result indicates that although the proposed method gains a low value of the set validity for 1-OACM, the result value is quite good compared to the expected value.

Table 5.5 The actual set validity, the expected set validity and their ratio, for various top-N rankings (soft validity)

Document representation	N	1-OACM			2-OACM			3-OACM		
		actual	expected	ratio	actual	expected	ratio	actual	expected	ratio
BXO	1000	45.47	0.07	676.13	92.56	1.43	64.85	98.47	9.88	9.97
	5000	29.31	0.07	401.43	79.96	1.55	51.64	96.52	10.67	9.04
	10000	24.49	0.07	327.52	74.62	1.59	47.07	95.22	10.89	8.74
	50000	19.29	0.11	180.36	73.40	2.25	32.60	95.77	14.88	6.44
	100000	19.51	0.13	145.03	72.08	2.78	25.96	94.87	16.98	5.59
UXO	1000	3.88	0.06	60.10	25.27	1.37	18.44	52.54	9.49	5.54
	5000	3.77	0.07	56.01	22.32	1.43	15.62	52.98	9.89	5.36
	10000	3.47	0.07	49.92	21.53	1.47	14.61	52.30	10.19	5.13
	50000	2.78	0.08	35.83	20.54	1.65	12.46	53.56	11.38	4.71
	100000	2.71	0.08	32.67	21.03	1.76	11.96	55.11	12.11	4.55

Table 5.6 The actual set validity, the expected set validity and their ratio, for various top-N rankings (hard validity)

Document representation	N	1-OACM			2-OACM			3-OACM		
		actual	expected	ratio	actual	expected	ratio	actual	expected	ratio
BXO	1000	43.95	0.06	754.31	92.28	1.24	74.33	98.47	8.79	11.20
	5000	24.88	0.05	502.81	77.68	1.06	73.26	96.44	7.75	12.44
	10000	19.33	0.05	402.36	71.30	1.03	69.26	95.10	7.54	12.61
	50000	6.36	0.02	381.96	57.10	0.37	154.06	94.70	3.35	28.30
	100000	3.67	0.01	309.01	47.84	0.27	179.06	93.28	2.49	37.45
UXO	1000	3.78	0.06	59.44	25.07	1.35	18.56	52.24	9.37	5.57
	5000	3.35	0.06	57.67	20.76	1.24	16.76	51.13	8.78	5.83
	10000	2.63	0.05	48.77	18.87	1.15	16.37	49.31	8.32	5.93
	50000	1.44	0.04	37.33	13.97	0.84	16.69	46.00	6.62	6.95
	100000	1.02	0.03	34.05	12.12	0.66	18.28	44.80	5.64	7.95

Table 5.7: Criteria for selecting pairs of documents as the sample document relations for hypothesis testing

Characteristic of citation relation that exists between two documents under			Number of selected document relations
1-OACM	2-OACM	3-OACM	
√	√	√	100
X	√	√	100
X	X	√	100
X	X	X	100

5.3 Results from Human Evaluation

Undoubtedly, the expected relations from word-based approach must be consistent with human intuition. As a word-based relation, the discovered docset will indicate a group of publications which contains similar topical content in some reasonable aspects. For example, a publication of "Revealing topic-based relationship among documents using association rule mining, Sriphaew K. and Theeramunkong T." may relate with other publications concerned with association rule mining in the aspect of an algorithm to be used, and may relate with the publications focusing on information retrieval in the aspect of finding relationship among documents. However, the judgment of relatedness among those publications is subjective to the user. It is varied depending on the user opinion. Therefore, the most reliable way to judge the effectiveness of discovered relations is by setting the human evaluation.

In this section, we present a set of experiments based on human evaluation to measure the relatedness among publications in the document relations. Based on our assumption that each discovered relation has high probability to exist in the citation graph, we then validate the co-relation between the property of measurement used for automatic evaluation and the subjective measurement given by humans. This observation has an interesting contribution in which we can avoid the labor-intensive and time-consuming task of human evaluation by using an alternative automatic evaluation method based on a citation graph for validating the discovered relations. Two studies of the observation are: (1) how important is the citation graph in representing the relatedness among publications based on human intuition, and (2) how likely is the relatedness given by humans on the document relations discovered from different document relations and the validity calculated from the automatic evaluation. Based on these issues, two experiments are implemented as follows.

5.3.1 Human Evaluation on Citation Information

Using citation information as a criteria for evaluating the discovered relations is a new approach of evaluation. However, belief in this criteria is still a question and affects to the trust in measurement based on it. To clarify this issue, we set an experiment to investigate the co-relation between the citation information and the desired relations that have been found by human. In the experiment, relatedness on the document relations when those relations are present or absent in the citation graph is assigned by human evaluators. The results can show the significance of difference in the relatedness between the document relations that are present in the citation graph and those not in the citation graph.

Table 5.8: Average relatedness (± standard deviation) given by four human evaluators on selected document relations

	Average relatedness		F-statistic	p-value
	Document relations exist in v-OACM	Document relations do *not* exist in v-OACM		
1-OACM	**78.40** (±23.60)	**43.33** (±35.19)	8.56	0.006
2-OACM	**74.20** (±25.67)	**30.00** (±30.86)	24.25	0.000
3-OACM	**66.13** (±28.55)	**10.00** (±17.88)	33.87	0.000

A set of document pairs is selected for this experiment. Based on the criteria for selecting the pairs in Table 5.7, the pairs of documents with different characteristics of citation relations under each OACM are randomly selected for evaluation. For example, in the first row of the table, one hundred pairs of documents which contain citation relations under 1-, 2- and 3-OACMs (δ^v among those two document in OACM is equal to 1) are randomly selected without replication. In total, four hundred pairs of documents are selected as the sample relations according to the given characteristics of citation relations under 1-OACM, 2-OACM and 3-OACM. To indicate the relatedness of each document relation, four experts holding Ph.D. degrees in computer science or engineering were asked to assign scores for those selected relations in random order and without repetitions. The experts carefully read the documents in a document relation one by one and assigned a score for their relatedness. The degree of relatedness is classified into three ordinal scales; 0% for 'not related', 50% for 'somewhat related', and 100% for 'related'. To determine the statistical significance of differences between automatic evaluation using a citation graph and human evaluation, we formulate the following null hypothesis:

$H0$: For a citation graph under each v-OACM, the human will *not* assign a degree of relatedness for a set of discovered relations in which their relations exist in the v-OACM significantly differs from another set in which their relations do not exist in the v-OACM.

Table 5.8 shows the summary of the average relatedness ± standard deviation (sd) given by the experts for each v-OACM. There are two groups of discovered relations to be considered, i.e., relations exist in the v-OACM and relations do not exist in the v-OACM. As shown in the table, the average of relatedness for relations that exist in the v-OACM are higher than that of relations that do not exist in the v-OACM for every case of v-OACM.

Furthermore, a one-way analysis of variance (ANOVA) technique is applied to test the statistical difference of relatedness given by humans between a group of relations that exist in the v-OACM and another group that does not exist in the v-OACM. The results of analysis are shown in Table 5.8 with the F-statistic and p-value. The F-statistic is the ratio of two estimations of a population variance based on the information in two groups. It provides a test for the statistical significance of the observed differences between the means of two groups (i.e., relations exist in the v-OACM and relations do not exist in the v-OACM). The p-value is a measure of how much evidence we have against the null hypothesis. From the

table, we can conclude against the null hypothesis that there are significant differences in the group of relations that exist in the v-OACM and another group that do not exist in the v-OACM at the 100% confidence intervals (p-value=0.000) for 2-OACM and 3-OACM and the 94% confidence intervals (p-value=0.006) for 1-OACM.

This conclusion confirms that our proposed evaluation method to use citation information under v-OACM is a comparable method to represent human intuition in assessing the relatedness of discovered relations. Some may argue that the sample relations for the significant analysis might be too small for making a conclusion. Therefore, we will also show the comparison between the result gained from human and automatic evaluation in the experiments to confirm a usefulness of the proposed automatic evaluation method.

5.3.2 Human Evaluation and Quality of Discovered Document Relations

In this experiment, we evaluate the quality of discovered docsets with the answers from human evaluators. Since it is a time consuming task to judge the quality of discovered docsets by hand, it is worth finding an automatic method to judge the quality of discovered docsets. Although the automatic evaluation method to use citation information was already presented in this work and used to evaluate the discovered docsets, it is necessary to confirm the conclusions made by the automatic evaluation again by using human evaluation. Therefore, we present this experiment.

Some discovered docsets from each top-N ranked docsets are systematically selected as representative samples. One docset from each chunk of one hundred ranked docsets is selected. Thus, we get 10, 50 and 100 docsets as the samples for top-1000, top-5000 and top-10000 ranked docsets, respectively. With the limitation of a labor-intensive task, we investigate the docsets discovered from two cases, i.e., 'BXO' and 'UXO'. Therefore, 320 docsets in total are selected for human judgment. The approach of indicating the relatedness to each docset is similar to the previous experiment where the degree of relatedness is one of the three scales; 0% for 'not related', 50% for 'somewhat related', and 100% for 'related'.

The percentages of average relatedness given by four experts are shown in Table 5.9. This result is consistent with the result from the proposed automatic evaluation method. Although the set validity cannot exactly reflect the relatedness of document relations, its value can distinguish the the performance difference between a high quality set and a low quality set of discovered relations. From the table, there are two interesting observations. First, the results from the bigram case ('BXO') are better than those from the unigram case ('UXO') for any top-N rankings. Second, the results show that better docsets can be discovered in the higher ranks rather than the lower ranks. Although only the average relatedness scores are shown here, the individual evaluation result obtained from each expert also preserves the performance order, i.e., 'BXO' has a higher relatedness score than 'UXO' and the higher rank has a higher relatedness score than the lower rank. These results support that the proposed evaluation method has high potential for using as an alternative method for evaluating the discovered docsets in order to avoid labor-intensive and time-consuming tasks in human evaluation. It is also noted that the set validity is lower than the relatedness score in every case. This phenomenon shows that our automatic evaluation method does not overestimate the quality of document relations.

Table 5.9: Average relatedness (± standard deviation) given by human evaluation and set 1-validity from automatic evaluation on samples of document relations discovered from 'BXO' and 'UXO' schemes

N	#Samples	Average relatedness from human evaluation		%Set 1-validity from automatic evaluation	
		BXO	UXO	BXO	UXO
1000	10	**77.08** (±15.05)	**21.25** (±10.31)	36.36	2.85
5000	50	**48.25** (±18.96)	**16.00** (±10.23)	29.31	2.00
10000	100	**34.46** (±18.04)	**12.17** (± 7.79)	19.66	1.33

5.3.3 Error Analysis

In the previous experiments, we found that some rules are invalid document relations with regard to some *v*-OACMs. To check their validity, we investigate the reason why this phenomenon occurs. We found that a set of documents in these invalid document relations contains one of the following characteristics.

- Those documents are the same articles which appear in various versions of publications or they are the minor change articles. They do not directly refer to each other. By evaluation, these relations are invalid when using the 1-OACM but they will be valid when evaluating by the 2-OACM and succeeding OACMs.
- Those documents have document relations but they do not link to each other or even share the same citing or cited articles, since they are published in the same year or same proceedings. We know that these documents should contain the document relations with each other because they appear in the same title of proceedings and contain quite similar contents.
- There are minor errors in the information pages downloaded from ACM Digital Library. In the information pages, some citations appearing in the publications are not given. Since we extract the citation matrix using the links from the citations in information page of each paper, the mistake is triggered by this missing information. Additionally, if the papers are not located in the ACM database, then there is no link between them. This situation rarely occurs in a set of discovered document relations.

Chapter 6

Experimental Results on Various Term Weighting

This chapter continues to investigate the term weighting schemes for improving the quality of discovered document relations. Several combinations of term weighting schemes which were successfully applied in text mining, information retrieval and text categorization approaches are examined in this work. A number of experiments are conducted to assess the document relations discovered from different term weighting schemes. The characteristics of document relations discovered from several term weighting schemes are also studied. Finally, the discussion on term weighting schemes which can help to enhance the quality of discovered document relations is summarized.

6.1 Experimental Settings

The three objectives of these experiments are (1) to study the quality of discovered relations when applying different term weighting schemes, (2) to measure the relative quality of discovered relations over the statistical estimation, and (3) to investigate the characteristics of discovered relations on several term weighting schemes. Towards the first objective, the term weighting schemes in Table 3.2 are explored for encoding the attribute-value database. Using the extended approach of frequent itemset mining, the document relations can be mined on an attribute-value database (the sample minings are illustrated in Section 3.5). The sets of discovered relations are evaluated to judge the performance of term weighting schemes. In the second objective, the expected validity is calculated to compare with the actual validity in order to show the relative quality of discovered document relations. As the last objective, the document lengths in some term weighting schemes are investigated to present the characteristics of discovered relations.

Besides term weighting, the term definition is also an important factor for representing the documents in an attribute-value database. With the conclusions from previous experiments, the best scheme for term definition; 'BXO': bigram, no-stemming and applying stopword removal, performs well in discovering high-quality document relations. We then explore these two term definition schemes together with the above term weighting schemes in this experiment to enhance the quality of discovered document relations.

The experiments are done on the same datasets as used in Chapter 5. Since the automatic evaluation was already verified in Chapter 5 as a potential method to measure the quality of discovered relations, the discovered relations will be evaluated based on the automatic evaluation. Although there are both soft and hard scoring methods, we can use either of them to judge the performance of term weighting schemes. As shown in the previous experiments,

if the soft validity of one document representation model is higher than another model, then the hard validity will perform the same. Therefore, the performance of any document representation model compared to other models can be relatively judged by either soft validity or hard validity. As a result, we will explore only the soft validity as an evaluation measure in this experiment.

6.2 Experimental Results

This section presents the quality of document relations discovered from various document representations generated by the combinations of term definition and term weighting schemes. The best term definition scheme, i.e., 'BXO': binary, non-stemming and applying stopword removal, is explored with ten term weighting schemes as shown in Table 3.2. The discovered docsets are ordered by their supports in descending order and the five top-*N* rankings, i,e, top-1000, top-5000, top-10000, top-50000 and top-100000, are selected for investigation. They are evaluated based on 1-, 2- and 3-OACMs with the soft validity calculation.

Table 6.1: Set 1-validity for various top-*N* rankings of discovered docsets when applying several term weighting schemes with 'BXO' as term definition.

N	1:bxx	1:bix	1:txx	1:tix	1:txc	1:txm	1:tic	1:tim	1:axx	1:aix
1000	45.47	54.30	26.90	45.64	29.85	23.77	52.74	53.90	46.76	55.17
5000	29.31	38.06	9.31	20.26	10.29	5.93	13.68	13.90	31.07	39.26
10000	24.49	32.20	8.95	18.87	9.71	5.05	11.59	8.71	25.60	33.75
50000	19.29	22.88	4.53	15.21	13.04	2.69	12.91	4.32	17.07	21.15
100000	19.51	23.35	3.17	14.42	12.10	2.86	12.90	3.74	11.47	16.69

6.2.1 Set Validity

Table 6.1 shows the set 1-validity of discovered docsets based on the 1-OACM using 'BXO' as the term definition. In the table, the notation is used to represent the evaluation of docsets under the specified OACM where those docsets are discovered from a specific term weighting scheme. For example, '1:bix' means the evaluation of docsets under 1-OACM where the docsets are discovered by encoding term weighting by the 'bix' scheme (binary term frequency, *idf* and no normalization). From the table, some interesting observations can be made. First, the term weightings with *idf* schemes provide the document relations with higher validity than those without *idf* schemes. This shows the potential of applying *idf* to discriminate the content of individual documents from the collection. Second, binary term frequency still performs well when compared with the term frequency, while the augmented normalized term frequency can gain higher validity in the higher ranks (small *N*). However, the validity of the augmented normalized term frequency cases drops more than the binary cases when the higher *N*'s are considered. One possible reason for this comes from the fact that most docsets in the lower rankings have the same supports when applying the augmented normalized term frequency. This situation means the docsets which contain less co-occurring terms can achieve support higher than the minimum support when the number of docsets is controlled as a threshold for mining. As a result, the discovered docsets have high probability to be invalid. For the third observation, the cosine normalization can help to discover the highly valid document relations in the higher ranks. However, there

is no certain conclusion for the performance of cosine normalization and maximum term frequency normalization since the results also depend on other term weighting factors. This can be observed by the comparison of validity in the four columns '1:txc', '1:txm', '1:tic' and '1:tim'.

Table 6.2: Set 2-validity for various top-*N* rankings of discovered docsets when applying several term weighting schemes with 'BXO' term definition.

N	2:bxx	2:bix	2:txx	2:tix	2:txc	2:txm	2:tic	2:tim	2:axx	2:aix
1000	92.56	94.74	72.26	86.09	76.51	67.09	92.12	90.69	92.56	95.11
5000	79.96	86.66	47.10	69.23	49.87	48.31	53.87	57.47	81.05	87.22
10000	74.62	81.60	43.66	66.05	48.69	48.33	48.37	51.32	76.58	82.18
50000	73.40	71.91	34.06	58.46	56.92	49.70	56.48	43.15	68.06	68.72
100000	72.08	76.01	30.95	55.80	54.61	47.39	57.65	42.35	56.24	63.12

Table 6.3: Set 3-validity for various top-*N* rankings of discovered docsets when applying several term weighting schemes with 'BXO' term definition.

N	3:bxx	3:bix	3:txx	3:tix	3:txc	3:txm	3:tic	3:tim	3:axx	3:aix
1000	98.47	98.57	91.81	94.34	93.48	93.79	97.50	97.93	98.47	98.75
5000	96.52	97.46	78.09	93.70	93.42	91.60	93.61	94.63	96.68	97.27
10000	95.22	96.27	76.73	93.99	93.47	91.20	93.00	94.42	95.72	96.07
50000	95.77	89.14	72.74	95.02	95.93	91.40	95.77	92.75	92.98	87.84
100000	94.87	94.45	71.87	93.84	96.37	92.15	96.52	90.55	89.45	90.03

Besides 1-OACM, the discovered docsets are also evaluated based on 2-OACM and 3-OACM as shown Table 6.2 and 6.3. From the tables, some observations can be made. First, similar to 1-OACM, *idf* schemes can improve the quality of discovered document relations, although it is only slightly improved for the evaluation based on 3-OACM. Second, the cases which apply binary term frequency, augmented normalized term frequency or *idf* with normalization can achieve approximately 90% set-validity for the docsets in the higher ranks. These show the variations of weighting schemes to represent the document contents. However, we can cnclude that the order of term weighting schemes for helping to improve performance of document relation discovery. By ordering from the weighting schemes which can enhance the performance, they can be ordered as follows: *idf*, binary frequency, augmented term frequency, cosine normalization, maximum term frequency normalization and term frequency.

Moreover, we also investigate the combinations of term weighting schemes and the unigram cases as term definition. Although the validity of those document representations performs in the same way as these bigram cases, their values are not higher than those from the bigram cases. Therefore, we then excluded the unigram cases from the exploration.

6.2.2 Set Validity vs. Expected Set Validity

The relative quality of discovered docsets against the expected validity is also investigated in the next experiment. Similar to Section 5.2.3, the expected set validity can be calculated using the base probabilities under 1-, 2-, and 3-OACMs. For brevity, only the investigations of 'bxx', 'bix', 'axx' and 'aix' are shown here. The actual set validity gained from the experiments, the expected set validity calculated from Equation 4.9 and their ratios are

displayed in Table 6.4. The ratio expresses the quality of the discovered docsets compared to its expected validity.

By comparing the result to the expected validity, the evaluations under different *v*-OACMs become comparable. From the tables, the quality of discovered docsets are still significantly better than the statistical estimation which is expressed by the ratio. Under the 1-, 2- and 3-OACMs, the ratios are approximately 100-800, 20-65 and 10-20, respectively. There are high variations of the ratio in lower-OACMs since the actual validity is highly varied in different ranks while the expected validity is constant. The order of their ratios is: 1-OACM $>$ 2-OACM $>$ 3-OACM, although the order of the set validity is: 3-OACM $>$ 2-OACM $>$ 1-OACM. The ratios of 1-OACM are approximately 4 to 12 times higher than those of 2-OACM, and the ratios of 2-OACM are approximately 4 to 7 times higher than those of 3-OACM. Moreover, the result also confirms that term weighting schemes which apply *idf* can gain higher ratios than those without *idf* in most cases. This states the outstanding performance of applying *idf* to represent the document content for document relation discovery against the other term weighting scheme.

Table 6.4: The actual set validity, the expected set validity and its ratio for the case ‘bxx’, ‘bix’, ‘axx’ and ‘aix’ as term weighting and ‘BXO’ as term definition.

Term weighting	N	1-OACM			2-OACM			3-OACM		
		actual	expected	ratio	actual	expected	ratio	actual	expected	ratio
bxx	1000	45.47	0.07	676.13	92.56	1.43	64.85	98.47	9.88	9.97
	5000	29.31	0.07	401.43	79.96	1.55	51.64	96.52	10.67	9.04
	10000	24.49	0.07	327.52	74.62	1.59	47.07	95.22	10.89	8.74
	50000	19.29	0.11	180.36	73.40	2.25	32.60	95.77	14.88	6.44
	100000	19.51	0.13	145.03	72.08	2.78	25.96	94.87	16.98	5.59
bix	1000	54.30	0.07	809.51	94.74	1.42	66.55	98.57	9.85	10.01
	5000	38.06	0.07	528.37	86.66	1.53	56.71	97.46	10.54	9.24
	10000	32.20	0.08	418.66	81.60	1.63	50.04	96.27	11.19	8.60
	50000	22.88	0.12	184.77	71.91	2.57	27.95	89.14	16.11	5.53
	100000	23.35	0.15	155.95	76.01	3.07	24.78	94.45	18.23	5.18
axx	1000	46.76	0.07	695.89	92.56	1.43	64.90	98.47	9.87	9.98
	5000	31.07	0.07	426.40	81.05	1.55	52.43	96.68	10.66	9.07
	10000	25.60	0.08	324.13	76.58	1.67	45.75	95.72	11.47	8.34
	50000	17.07	0.13	132.61	68.06	2.68	25.44	92.98	16.71	5.57
	100000	11.47	0.18	64.65	56.24	3.57	15.74	89.45	20.04	4.46
aix	1000	55.17	0.07	823.93	95.11	1.42	66.93	98.75	9.83	10.04
	5000	39.26	0.07	549.94	87.22	1.51	57.58	97.27	10.46	9.30
	10000	33.75	0.08	445.36	82.18	1.61	51.14	96.07	11.05	8.69
	50000	21.15	0.12	174.58	68.72	2.51	27.34	87.84	15.73	5.58
	100000	16.69	0.16	106.00	63.12	3.20	19.72	90.03	18.59	4.84

6.2.3 Characteristic of Discovered Document Relations

A set of discovered docsets consists of many docsets with different sizes. With the bigger docset, an increasing number of articles causes a lower value of validity. In practice, the bigger the docset is, the lower the value of validity becomes. Table 6.5 shows the number of docsets of each length, where several term weighting schemes are applied with 'BXO' as term definition in the top-100000 ranked docsets. From the table, three interesting characteristics are observed: (1) an *idf* scheme does not affect the length of document relations, (2) a *tf* scheme provides a larger number of bigger docsets than an augmented normalized term frequency scheme, (3) an augmented normalized term frequency scheme also produces a larger number of bigger docsets than a binary frequency scheme, although those two schemes are comparable in the quality of document relations, and (4) there is no significant effect from applying cosine or maximum *tf* normalizations. By these observations, the preference of high-quality document relations in the aspect of docset length can be selected by applying different term weighting schemes. It can be applied in the applications when the lengths of document relations are concerned.

Table 6.5: The number of docsets of each length where several term weighting schemes are applied with 'BXO' as term definition in the top-100000 ranked docsets

Docset length	bxx	bix	txx	tix	txc	txm	tic	tim	axx	aix
2	40,870	37,070	2,035	2,446	2,318	2,741	2,769	4,374	26,308	38,132
3	30,679	21,608	2,032	1,498	1,860	2,553	1,265	1,498	17,580	20,691
4	10,759	15,686	4,386	3,285	3,836	6,092	2,940	2,980	15,023	14,006
5	8,004	11,322	9,775	7,606	7,660	10,777	6,497	6,822	13,638	10,089
6	5,266	7,357	17,718	14,358	12,892	15,622	11,503	12,640	11,488	7,361
7	2,835	4,091	22,315	21,834	17,748	18,408	16,426	18,305	7,969	5,261
8	1,168	1,881	14,975	21,057	19,825	17,696	18,876	20,882	4,754	2,931
9	347	698	12,331	11,982	17,467	13,349	16,184	18,855	2,240	1,076
10	66	217	8,038	8,486	9,036	7,598	12,100	10,026	782	355
11	6	58	4,126	4,699	4,735	3,577	7,053	2,536	188	84
12		11	1,647	1,985	1,911	1,242	3,120	852	28	13
13		1	498	615	574	298	1,009	199	2	1
14			108	131	121	44	225	29		
15			15	17	16	3	31	2		
16			1	1	1		2			

Chapter 7

Conclusions and Future Work

This chapter summarizes all research works done in this thesis. The key contributions are listed to present the achievement and impact of this work. Some recommendations for open issues and directions are also discussed for future research.

7.1 Summary

The thesis presents a new approach to discover document relations using extended frequent itemset mining technique including a method to use citation information of research publications as a source for evaluating the discovered document relations. Five contributions of this work are as follows:

1. The thesis presented a method to discover the document relations among the collection of scientific publications using the extended frequent itemset mining approach. The extended approach is general enough to mine on both conventional transactional database with boolean (binary) values and extended transactional database with real values. With this extension, the high-quality document relations can be discovered.

2. The thesis studied the approach of encoding document representation with several term definition and term weighting schemes and presented the quality of discovered relations for each document representation model. For the term definition scheme, bigram cases provide better document relations than unigram cases and applying stopword removal is preferable while the stemming slightly affects the quality. For the term weighting scheme, *idf* dramatically improves the quality of document relations, binary term frequency is comparable to the augmented normalized term frequency and both perform better than the term frequency, and there is no significant difference between cosine and maximum weight normalization.

3. The thesis explored the validity with both soft/hard scorings including the evaluation based on either direct or indirect citations (different v-OACM). The soft validity is always higher than or equal to the hard validity since the former is obtained by less restrictive evaluation than the latter. Both can be used as the measurement to reflect the quality of discovered relations. The discovered relations are more valid to the case of both direct and indirect citations (2- and 3-OACMs) than the case of the direct citations (1-OACM) alone.

4. The thesis gave an analysis on the actual validity and the expected validity estimated from generative probability on v-OACM. To make the evaluation criteria impartial,

the actual validity is compared to its expected validity which was calculated from the statistical estimation regardless of the v-OACM. The result indicates that although the proposed method gains low set validity for 1-OACM, the result is quite good compared to the expected validity.

5. The thesis compared the results from the proposed automatic evaluation with the results from human evaluation to confirm the usability of the proposed automatic evaluation. The result confirms that the proposed automatic evaluation using citation information under v-OACM is comparable to human intuition in assessing the relatedness of discovered relations, and provides the consistent conclusions with the human evaluation.

7.2 Future Study

As the future study on document relation discovery, the following topics are interesting for further exploration. First, instead of frequent itemsets for representing document relations, the association rule is another interesting knowledge for expressing document relations. By this, we need to consider the direction of relations between the documents. Second, a hybrid approach which utilizes both terms in documents and citations among documents for discovering document relations is also valuable for further investigation. It is also interesting to apply this approach to the web data collections and utilize the hyperlinks between web pages as information for retrieving high-quality document relations. The third topic concerns granularity of relations among documents. Instead of the relations between full documents, the fine-grained relations among portions of documents or the more general relations between groups of documents are an interesting knowledge to be discovered. This approach can be implemented by the method of generalized frequent itemset mining where several parameters need to be investigated.

Bibliography

[Agrawal et al., 1993a] Agrawal, R., Imielinski, T., and Swami, A. N. (1993a). Mining association rules between sets of items in large databases. In Buneman, P. and Jajodia, S., editors, *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, pages 207–216, Washington, D.C.

[Agrawal et al., 1993b] Agrawal, R., Imielinski, T., and Swami, A. N. (1993b). Mining association rules between sets of items in large databases. In Buneman, P. and Jajodia, S., editors, *Proc. of the 1993 ACM SIGMOD Int'l Conf. on Management of Data*, pages 207–216, Washington, D.C.

[Agrawal et al., 1996] Agrawal, R., Mannila, H., Srikant, R., Toivonen, H., and Verkamo, A. I. (1996). Fast discovery of association rules. pages 307–328.

[Agrawal and Srikant, 1994] Agrawal, R. and Srikant, R. (1994). Fast algorithms for mining association rules. In Bocca, J. B., Jarke, M., and Zaniolo, C., editors, *Proceedings of the 20th International Conference on Very Large Data Bases, VLDB*, pages 487–499. Morgan Kaufmann.

[Allan, 1997] Allan, J. (1997). Building hypertext using information retrieval. *Informational Processing and Management*, 33(2):145–159.

[An et al., 2004] An, Y., Janssen, J., and Milios, E. E. (2004). Characterizing and mining the citation graph of the computer science literature. *Knowl. Inf. Syst.*, 6(6):664–678.

[B.A.Davey and H.A.Priestley, 2002] B.A.Davey and H.A.Priestley (2002). *Introduction to Lattices and Order*. Cambridge University Press, second edition.

[Baeza-Yates and Ribeiro-Neto, 1999] Baeza-Yates, R. A. and Ribeiro-Neto, B. A. (1999). *Modern Information Retrieval*. ACM Press / Addison-Wesley.

[Bastide et al., 2000] Bastide, Y., Pasquier, N., Taouil, R., Stumme, G., and Lakhal, L. (2000). Mining minimal non-redundant association rules using frequent closed itemsets. *Lecture Notes in Computer Science*, 1861:972–986.

[Beil et al., 2002] Beil, F., Ester, M., and Xu, X. (2002). Frequent term-based text clustering. In *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on knowledge discovery and data mining*, pages 436–442, New York, NY, USA. ACM Press.

[Bergmark, 2000] Bergmark, D. (2000). Automatic extraction of reference linking information from online documents. Technical report, Cornell University, Ithaca, NY, USA.

[Bjorneborn, 2004] Bjorneborn, L. (2004). *Small-World Link Structures across an Academic Web Space: A Library and Information Science Approach*. PhD thesis, Royal School of Library and Information Science, Denmark.

[Buckley, 1993] Buckley, C. (1993). The importance of proper weighting methods. In *HLT '93: Proceedings of the workshop on Human Language Technology*, pages 349–352, Morristown, NJ, USA. Association for Computational Linguistics.

[Cai et al., 1998] Cai, C. H., Fu, A. W. C., Cheng, C. H., and Kwong, W. W. (1998). Mining association rules with weighted items. In *IDEAS '98: Proceedings of the 1998 International Symposium on Database Engineering & Applications*, page 68, Washington, DC, USA. IEEE Computer Society.

[Chen, 1999] Chen, C. (1999). Visualising semantic spaces and author co-citation networks in digital libraries. *Information Processing and Management*, 35(3):401–420.

[Clifton and Cooley, 1999] Clifton, C. and Cooley, R. (1999). Topcat: Data mining for topic identification in a text corpus. In *Principles of Data Mining and Knowledge Discovery*, pages 174–183.

[da Silva et al., 2001] da Silva, J. F., Mexia, J., Coelho, C. A., and Lopes, J. G. P. (2001). Document clustering and cluster topic extraction in multilingual corpora. In *Proceedings of the 2001 IEEE International Conference on Data Mining*, pages 513–520, 29 November - 2 December 2001, San Jose, California, USA. IEEE Computer Society.

[Egghe and Rousseau, 2002] Egghe, L. and Rousseau, R. (2002). Co-citation, bibliographic coupling and a characterization of lattice citation networks. *Scientometrics*, 55(3):349–361.

[Ehrler et al., 2005] Ehrler, F., Geissbuhler, A., Jimeno, A., and Ruch, P. (2005). Data-poor categorization and passage retrieval for gene ontology annotation in swiss-prot. *BMC Bioinformatics*, 6(1):S23.

[Ertoz et al., 2003] Ertoz, L., Steinbach, M., and Kmar, V. (2003). Finding clusters of different sizes, shapes, and densities in noisy, high dimensional data. In *Proc. of the 3rd SIAM International Conference on Data Mining*, pages 326–331, San Francisco, CA, USA.

[Faloutsos and Oard, 1995] Faloutsos, C. and Oard, D. W. (1995). A survey of information retrieval and filtering methods. Technical Report CS-TR-3514, Electrical Engineering, University of Maryland, College Park, MD 20742.

[Feldman et al., 1998] Feldman, R., Fresko, M., Kinar, Y., Lindell, Y., Liphstat, O., Rajman, M., Schler, Y., and Zamir, O. (1998). Text mining at the term level. In *Principles of Data Mining and Knowledge Discovery*, pages 65–73.

[Furuta et al., 1989] Furuta, R. K., Plaisan, C., and Shneiderman, B. (1989). A spectrum of automatic hypertext constructions. *Hypermedia*, 1(2):179–195.

[Ganiz et al., 2005] Ganiz, M. C., Pottenger, W. M., and Janneck, C. D. (2005). Recent advances in literature based discovery. Technical Report LU-CSE-05-027., Lehigh University.

[Ganter and Wille, 1997] Ganter, B. and Wille, R. (1997). *Formal Concept Analysis: Mathematical Foundations*. Springer-Verlag New York, Incorporated.

[Garfield, 1972] Garfield, E. (1972). Citation analysis as a tool in journal evaluation. *Science*, 178(4060)(60):471–479.

[Garfield, 1995] Garfield, E. (1995). Citation indexes for science.

[Garfield, 2001] Garfield, E. (2001). From bibliographic coupling to co-citation analysis via algorithmic historio-bibliography.

[Glenisson et al., 2003] Glenisson, P., Mathys, J., and Moor, B. D. (2003). Meta-clustering of gene expression data and literature-based information. *SIGKDD Explor. Newsl.*, 5(2):101–112.

[Gordon and Dumais, 1998] Gordon, M. and Dumais, S. (1998). Using latent semantic indexing for literature based discovery. *Journal of the American Society for Information Science*, 49(8):674–685.

[Gordon and Lindsay, 1996] Gordon, M. D. and Lindsay, R. K. (1996). Toward discovery support systems: a replication, re-examination, and extension of *S*wanson's work on literature-based discovery of a connection between *R*aynaud's and fish oil. *Journal of the American Society for Information Science*, 47(2):116–128.

[Han and Fu, 1999] Han, J. and Fu, Y. (1999). Mining multiple-level association rules in large databases. *Knowledge and Data Engineering*, 11(5):798–804.

[Han et al., 2000] Han, J., Pei, J., and Yin, Y. (2000). Mining frequent patterns without candidate generation. In Chen, W., Naughton, J., and Bernstein, P. A., editors, *2000 ACM SIGMOD Intl. Conference on Management of Data*, pages 1–12. ACM Press.

[Han et al., 2004] Han, J., Pei, J., Yin, Y., and Mao, R. (2004). Mining frequent patterns without candidate generation: A frequent-pattern tree approach. *Data Min. Knowl. Discov.*, 8(1):53–87.

[He and Hui, 2002] He, Y. and Hui, S. C. (2002). Mining a web citation database for author co-citation analysis. *Information Processing and Management*, 38(4):491–508.

[Hetzler, 1997] Hetzler, E. G. (1997). Beyond word relations - sigir '97 workshop. *SIGIR Forum*, 31(2):28–33.

[Hipp et al., 1998] Hipp, J., Myka, A., Wirth, R., and G*ü*ntzer, U. (1998). A new algorithm for faster mining of generalized association rules. In *Proceedings of the 2nd European Conference on Principles of Data Mining and Knowledge Discovery (PKDD '98)*, pages 74–82, Nantes, France.

[Hung and Wermter, 2003] Hung, C. and Wermter, S. (2003). A dynamic adaptive self-organising hybrid model for text clustering. In *Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM 2003), 19-22 December 2003, Melbourne, Florida, USA*, pages 75–82. IEEE Computer Society.

[Hwang and Lim, 2002] Hwang, S.-Y. and Lim, E.-P. (2002). A data mining approach to new library book recommendations. In *Lecture Notes in Computer Science ICADL 2002*, volume 2555, pages 229–240, Singapore.

[Jin et al., 2001] Jin, R., Falusos, C., and Hauptmann, A. G. (2001). Meta-scoring: automatically evaluating term weighting schemes in ir without precision-recall. In *SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 83–89, New York, NY, USA. ACM Press.

[Jones, 1972] Jones, K. S. (1972). A statistical interpretation of term specificity and its application in retrieval. *Documentation*, 28(1):11–21.

[Jones and van Rijsbergen, 1975] Jones, K. S. and van Rijsbergen, C. (1975). Report on the need for and provision of an ideal information retrieval test collection.

[Jones and Willett, 1997] Jones, K. S. and Willett, P., editors (1997). *Readings in information retrieval*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

[Kessler, 1963] Kessler, M. M. (1963). Bibliographic coupling between scientific papers. *American Documentation*, 14:10–25.

[Kleinberg, 1999] Kleinberg, J. M. (1999). Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632.

[Klemettinen et al., 1994] Klemettinen, M., Mannila, H., Ronkainen, P., Toivonen, H., and Verkamo, A. I. (1994). Finding interesting rules from large sets of discovered association rules. In Adam, N. R., Bhargava, B. K., and Yesha, Y., editors, *Third International Conference on Information and Knowledge Management (CIKM'94)*, pages 401–407. ACM Press.

[Kostoff et al., 2001] Kostoff, R. N., del Rio, J. A., Humenik, J. A., Garcia, E. O., and Ramirez, A. M. (2001). Citation mining: integrating text mining and bibliometrics for research user profiling. *Journal of the American Society of Information Science*, 52(13):1148–1156.

[Lawrence et al., 1999] Lawrence, S., Giles, C. L., and Bollacker, K. (1999). Digital libraries and autonomous citation indexing. *IEEE Computer*, 32(6):67–71.

[Lelu, 1991] Lelu, A. (October 1991). Automatic generation of 'hyper-paths in information retrieval systems: A stochastic and an incremental algorithms. In *Proceedings of ACM SIGIR '91*, pages 326–335, Chicago, Illinois.

[Lin et al., 2003] Lin, X., White, H. D., and Buzydlowski, J. (2003). Real-time author co-citation mapping for online searching. *Information Processing and Management*, 39(5):689–706.

[Lindsay and Gorden, 1999] Lindsay, R. and Gorden, M. (1999). Literature-based discovery by lexical statistics. *Journal of the American Society for Information Science*, 50(7):574–587.

[Lu et al., 2006] Lu, W., Janssen, J., Milios, E., Japkowicz, N., and Zhang, Y. (2006). Node similarity in the citation graph. *Knowledge and Information Systems*, 11(1):105–129.

[Lui and Chung, 2000] Lui, C. L. and Chung, F. L. (2000). Discovery of generalized association rules with multiple minimum supports. In *Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery (PKDD '2000)*, pages 510–515, Lyon, France.

[McCallum, 1996] McCallum, A. K. (1996). Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering.

[Michail, 2000] Michail, A. (2000). Data mining library reuse patterns using generalized association rules. In *International Conference on Software Engineering*, pages 167–176.

[Mizzaro, 1999] Mizzaro, S. (1999). Measuring the agreement among relevance judges.

[Moon and Singh, 2005] Moon, N. and Singh, R. (2005). Experiments in text-based mining and analysis of biological information from medline on functionally-related genes. In *ICSENG '05: Proceedings of the 18th International Conference on Systems Engineering*, pages 326–331, Washington, DC, USA. IEEE Computer Society.

[Nahm and Mooney, 2000] Nahm, U. Y. and Mooney, R. J. (2000). A mutually beneficial integration of data mining and information extraction. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*, pages 627–632. AAAI Press/The MIT Press.

[Nanba et al., 2000] Nanba, H., Kando, N., and Okumura, M. (2000). Classification of research papers using citation links and citation types: Towards automatic review article generation. In *Proceedings of the American Society for Information Science (ASIS) / the 11th SIG Classification Research Workshop, Classification for User Support and Learning*, pages 117–134, Chicago, USA. Morgan Kaufmann Publishers, San Francisco, US.

[Nigam et al., 2000] Nigam, K., McCallum, A., Thrun, S., and Mitchell, T. (2000). Text classification from labeled and unlabeled documents using em. *Machine Learning*, 39(2/3):103–134.

[Padmanabhan and Tuzhilin, 1999] Padmanabhan, B. and Tuzhilin, A. (1999). Unexpectedness as a measure of interestingness in knowledge discovery. *Decision Support Systems*, 27(3):303–318.

[Page et al., 1998] Page, L., Brin, S., Motwani, R., and Winograd, T. (1998). The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project.

[Pasquier et al., 1999] Pasquier, N., Bastide, Y., Taouil, R., and Lakhal, L. (1999). Discovering frequent closed itemsets for association rules. *Lecture Notes in Computer Science*, 1540:398–416.

[Pietracaprina and Zandolin, 2003] Pietracaprina, A. and Zandolin, D. (2003). Mining frequent itemsets using patricia tries.

[Porter, 1980] Porter, M. (1980). An algorithm for suffix stripping. Progam, vol.14, no. 3.

[Pratt et al., 1999] Pratt, W., Hearst, M., and Fagan, L. (1999). A knowledge-based approach to organizing retrieved documents. In *Proc. of the Sixteenth National Conference on Artificial Intelligence (AAAI-99)*, pages 80–85, Orlando.

[Pratt and Yetisgen-Yildiz, 2003] Pratt, W. and Yetisgen-Yildiz, M. (2003). Litlinker: capturing connections across the biomedical literature. In *K-CAP '03: Proceedings of the 2nd international conference on Knowledge capture*, pages 105–112, New York, NY, USA. ACM Press.

[Rahal et al., 2006] Rahal, I., Ren, D., Wu, W., Denton, A., Besemann, C., and Perrizo, W. (2006). Exploiting edge semantics in citation graphs using efficient, vertical arm. *Knowledge and Information Systems*, 10(1):57–91.

[Rajman and Besançon, 1998] Rajman and Besançon (1998). Text mining — knowledge extraction from unstructured textual data. In *6th Conference of International Federation of Classification Societies (IFCS-98), Rome*.

[Redner, 1998] Redner, S. (1998). How popular is your paper? an empirical study of the citation distribution. *The European Physical Journal B*, 4:131.

[Rocchio, 1971] Rocchio, J. J. (1971). *Relevance Feedback in Information Retrieval*, chapter 14, pages 313–323. Prentice-Hall, Englewood Clis, NJ.

[Rosch, 1978] Rosch, E. (1978). *Principles of Categorization*, pages 27–48. John Wiley & Sons Inc.

[Rousseau and Zuccala, 2004] Rousseau, R. and Zuccala, A. (2004). A classification of author co-citations: definitions and search strategies. *J. Am. Soc. Inf. Sci. Technol.*, 55(6):513–529.

[Ruch, 2006] Ruch, P. (2006). Automatic assignment of biomedical categories: toward a generic approach. *Bioinformatics*, 22(6):658–664.

[Salton and Buckley, 1988] Salton, G. and Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5):513–523.

[Salton and Buckley, 1991] Salton, G. and Buckley, C. (October 1991). Automatic text structuring and retrieval - experiments in automatic encyclopedia searching. In *Proceedings of ACM SIGIR '91*, pages 21–30, Chicago, Illinois.

[Salton and McGill, 1983] Salton, G. and McGill, M. (1983). *Introduction to Modern Information Retrieval*. McGraw-Hill.

[Salton and McGill, 1986] Salton, G. and McGill, M. J. (1986). *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., New York, NY, USA.

[Salton et al., 1975] Salton, G., Wong, A., and Yang, C. S. (1975). A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620.

[Salton and Yang, 1973] Salton, G. and Yang, C. (1973). On the specification of term values in automatic indexing. *Documentation*, 29:351–372.

[Shintani and Kitsuregawa, 1998] Shintani, T. and Kitsuregawa, M. (1998). Parallel mining algorithms for generalized association rules with classification hierarchy. In *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data*, pages 25–36.

[Silberschatz and Tuzhilin, 1995] Silberschatz, A. and Tuzhilin, A. (1995). On subjective measures of interestingness in knowledge discovery. In *Knowledge Discovery and Data Mining*, pages 275–281.

[Silberschatz and Tuzhilin, 1996] Silberschatz, A. and Tuzhilin, A. (1996). What makes patterns interesting in knowledge discovery systems. *Ieee Trans. On Knowledge And Data Engineering*, 8:970–974.

[Small, 1973] Small, H. (1973). Co-Citation in the scientific literature: a new measure of the relationship between documents. *Journal of the American Society for Information Science*, 42:676–684.

[Srikant and Agrawal, 1997] Srikant, R. and Agrawal, R. (1997). Mining generalized association rules. *Future Generation Computer Systems*, 13(2–3):161–180.

[Sriphaew and Theeramunkong, 2002] Sriphaew, K. and Theeramunkong, T. (2002). A new method for finding generalized frequent itemsets in generalized association rule mining. In Corradi, A. and Daneshmand, M., editors, *Proceedings of the Seventh International Symposium on Computers and Communications*, pages 1040–1045, Taormina-Giardini Naxos, Italy.

[Swanson, 1986] Swanson, D. (1986). Fish oil, Raynaud's syndrome, and undiscovered public knowledge. *Perspectives in Biology and Medicine*, 30(1):7–18.

[Swanson, 1990] Swanson, D. (1990). Medical literature as a potential source of new knowledge. *Bulletin of the Medical Library Association*, 78(1):29–37.

[Tao et al., 2003] Tao, F., Murtagh, F., and Farid, M. (2003). Weighted association rule mining using weighted support and significance framework. In *KDD '03: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 661–666, New York, NY, USA. ACM Press.

[Theeramunkong, 2004] Theeramunkong, T. (2004). Applying passage in web text mining. *Int. J. Intell. Syst.*, 19(1-2):149–158.

[Valtchev et al., 2000] Valtchev, P., Missaoui, R., and Lebrun, P. (2000). A fast algorithm for building the hasse diagram of a galois lattice.

[Van Rijsbergen, 1979] Van Rijsbergen, C. J. (1979). *Information Retrieval, 2nd edition*. Dept. of Computer Science, University of Glasgow.

[White and McCain, 1989] White, H. and McCain, K. (1989). Bibliometrics. In Williams, M., editor, *Annual review on information science and technology*, pages 119–186, Amsterdam, Netherlands. Elsevier Science Publishers.

[White, 2003] White, H. D. (2003). Pathfinder networks and author co-citation analysis: a remapping of paradigmatic information scientists. *Journal of the American Society for Information Science and Technology*, 54(5):423–434.

[Wilkinson and Smeaton, 1999] Wilkinson, R. and Smeaton, A. F. (1999). Automatic link generation. *ACM Computing Survey*, 31(4es):27.

[Wu and Salton, 1981] Wu, H. and Salton, G. (1981). A comparison of search term weighting: term relevance vs. inverse document frequency. In *SIGIR '81: Proceedings of the 4th annual international ACM SIGIR conference on Information storage and retrieval*, pages 30–39, New York, NY, USA. ACM Press.

[Yang, 1999] Yang, Y. (1999). An evaluation of statistical approaches to text categorization. *Information Retrieval*, 1(1/2):69–90.

[Yu et al., 1982] Yu, C. T., Lam, K., and Salton, G. (1982). Term weighting in information retrieval using the term precision model. *Journal of ACM*, 29(1):152–170.

[Yun and Leggett, 2006] Yun, U. and Leggett, J. J. (2006). Wip: mining weighted interesting patterns with a strong weight and/or support affinity. In *Online Proceedings of 2006 SIAM Conference on Data Mining*, pages 623–627, Bathesda, Maryland, USA. IEEE Computer Society.

[Zaki, 2000] Zaki, M. J. (2000). Generating non-redundant association rules. In *Knowledge Discovery and Data Mining*, pages 34–43.

[Zaki and Hsiao, 2002] Zaki, M. J. and Hsiao, C.-J. (2002). CHARM: An efficient algorithm for closed itemset mining. In Grossman, R., Han, J., Kumar, V., Mannila, H., and Motwani, R., editors, *Proceedings of the Second SIAM International Conference on Data Mining*, Arlington VA.

[Zaki and Ogihara, 1998] Zaki, M. J. and Ogihara, M. (1998). Theoretical foundations of association rules. In *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data*, pages 7:1–7:8, Seattle, Washington.

[Zhang and Nguyen, 2005] Zhang, J. and Nguyen, T. N. (2005). A new term significance weighting approach. *Journal of Intelligent Information Systems*, 24(1):61–85.

Appendix A

Generalized Frequent Itemset Mining

This section presents the research that focuses on studying algorithms for generalized frequent itemset mining and generalized closed frequent itemset mining when there is a taxonomy presented on the items. The proposed algorithms are general and efficient enough to apply for any works which can be formulated as a problem of generalized frequent itemset mining. As one approach, the problem of document relation discovery can be viewed as a problem of generalized frequent itemset mining where the documents are partitioned into small portions or grouped as classes and we can find the relations among portions of documents or classes. Although it is possible to apply the concept of generalized frequent itemset mining for document relation discovery, it was left as our future work.

Generalized Association Rules and Generalized Frequent Itemsets

With the presence of a taxonomy, the formal problem description of generalized association rules is different from earlier works on association rule mining [Agrawal et al., 1993a, Agrawal and Srikant, 1994]. For clarity, all explanations in the section are illustrated using an example shown in Figure A.1.

Let $\mathcal{T}$ be a taxonomy, a directed acyclic graph on items, which represents *is-a* relationship by edges, e.g. Figure A.1C. The items in $\mathcal{T}$ are composed of a set of leaf items (I_L) and a set of non-leaf items (I_{NL}). Let $I = \{i_1, i_2, ..., i_m\}$ be a set of distinct items where $I = I_L \cup I_{NL}$, and let $T = \{1, 2, ..., n\}$ be a set of transaction identifiers (*tids*). In this example, $I_L = \{A, B, C, D, E\}$, $I_{NL} = \{U, V, W\}$, $I = I_L \cup I_{NL} = \{A, B, C, D, E, U, V, W\}$, and $T = \{1, 2, 3, 4, 5, 6\}$. A subset of I is called an *itemset* and a subset of T is called a *tidset*. Normally, a transactional database is represented in the horizontal database format, where each transaction corresponds to an itemset (e.g. Figure A.1A). An alternative to the horizontal database format is the vertical database format, where each item corresponds to a tidset which contains that item (e.g. Figure A.1B). Note that the original database contains only leaf items. It is possible to represent an original vertical database by extending it to cover non-leaf items where a transaction of each leaf item also supports its ancestor items from taxonomy (e.g. Figure A.1D). Let the binary relation $\delta \subseteq I \times T$ be an extended database. For any $x \in I$ and $y \in T$, $x\delta y$ can be denoted when x is related to y in the database (called x is supported by y). Here, except for the elements in I, lower case letters are used to denote items and upper case letters for itemsets.

For $\hat{x}, x \in I$, $\hat{x}$ is an *ancestor* of x (conversely x is a *descendant* of $\hat{x}$) when there is a path from $\hat{x}$ to x in $\mathcal{T}$. For any $x \in I$, a set of all its ancestors (descendants) is denoted by $\mathcal{ANC}(x)$

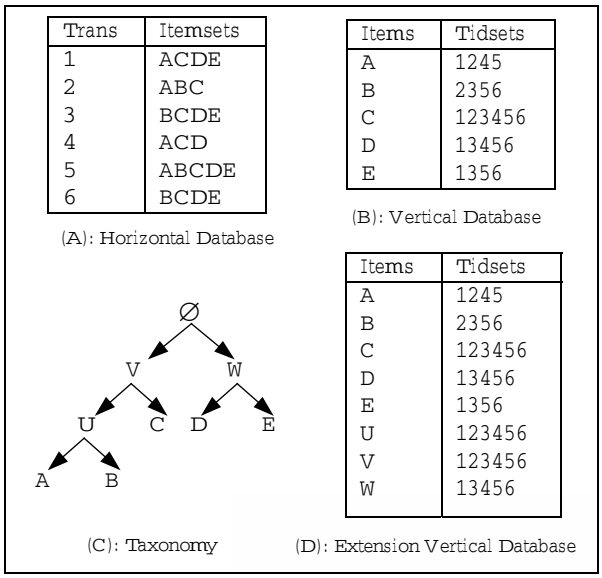


Figure A.1 An example of databases and taxonomy

($\mathcal{DES}(x)$). For example, $\mathcal{ANC}(B) = \{U, V\}$ and $\mathcal{DES}(W) = \{D, E\}$.

A generalized itemset G is an itemset each element of which is not an ancestor of the others, $G = \{i \in I | \forall j \in G, i \notin \mathcal{ANC}(j)\}$. For example, $\{A, B\}$ (AB for short), $\{A, W\}$ (AW for short) are generalized itemsets. Let $I_G = \{G_1, G_2, ..., G_l\}$ be a finite set of all generalized itemsets. Note that, for $1 \leq i \leq l$, $G_i \subseteq I$ and $I_G \subseteq \mathcal{P}(I)$. The support of G, denoted by $\sigma(G)$, is defined by a percentage of the number of transactions in which G occurs as a subset to the total number of transactions, thus $\sigma(G) = |t(G)|/|T|$. Any G is called *generalized frequent itemset* (GFI) when its support is at least a user-specified *minimum support* (*minsup*) threshold.

In GARM, a *meaningful* rule is an implication of the form $\mathcal{R} : G_1 \rightarrow G_2$, where $G_1, G_2 \in I_G, G_1 \cap G_2 = \emptyset$, and no item in G_2 is an ancestor of any items in G_1. For example, $A \rightarrow C$ and $U \rightarrow C$ are meaningful rules, while $A \rightarrow UC$ is a *meaningless* rule because its support is redundant with $A \rightarrow C$. The *support* of a rule $G_1 \rightarrow G_2$, defined as $\sigma(G_1 \cup G_2) = |t(G_1 \cup G_2)|/|T| = |t(G_1) \cap t(G_2)|/|T|$, is the percentage of the number of transactions containing both G_1 and G_2 to the total number of transactions. The *confidence of a rule*, defined as $\sigma(G_1 \cup G_2)/\sigma(G_1)$, is the conditional probability that a transaction contains G_2, given that it contains G_1. For example, the support of $A \rightarrow C$ is $\sigma(A \cup C) = |t(A) \cap t(C)|/|T| = |1245|/6$ = 4/6 or 67% and the confidence is $\sigma(A \cup C)/\sigma(A)$ = 1 or 100%. The meaningful rule is called a *generalized association rule* (GAR) when its confidence is at least a user-specified *minimum confidence* (*minconf*) threshold.

The task of GARM is to discover all GARs the supports and confidences of which are at least *minsup* and *minconf*, respectively.

Two Relationships on Generalized Itemsets

This section introduces two relationships, i.e. subset-superset and ancestor-descendant relationships, based on lattice theory. For more details about lattice theory, the reader can

refer to [B.A.Davey and H.A.Priestley, 2002]. To construct the generalized itemset lattice in GARM, we adapt the formal concept analysis [Ganter and Wille, 1997] and itemset lattice in ARM [Zaki and Ogihara, 1998]. Similar to ARM, GARM occupies the subset-superset relationship which represents a lattice of generalized itemsets. As the second relationship, an ancestor-descendant relationship is originally introduced in this work to represent a set of *k*-generalized itemset taxonomies.

Subset-Superset Relationship: Lattice of Generalized Itemsets

Definition 1 (Subset-superset relationship) *Let a binary relation $\delta_S \subseteq \mathcal{P}(I) \times \mathcal{P}(I)$ be the subset-superset relationship. For any $X_1, X_2 \in I_G$, $X_1 \delta_S X_2$ is denoted when X_1 is a subset of X_2 (X_2 is a superset of X_1).*

Definition 2 (Lattice of generalized itemsets) *The lattice of generalized itemsets is the partial order specified by a subset-superset relationship δ_S, where the meet is given by the set intersection on generalized itemsets, and the join is given by the set union on generalized itemsets as follows. For any $X_1, X_2 \in I_G$,*

$$\begin{aligned} \textit{Meet} &: X_1 \wedge X_2 = (X_1 \cap X_2) \\ \textit{Join} &: X_1 \vee X_2 = (X_1 \cup X_2) \end{aligned}$$

Ancestor-Descendant Relationship: *k*-Generalized Itemset Taxonomies

Definition 3 (Ancestor-Descendant relationship) *Let a binary relation $\delta_{\mathcal{A}} \subseteq \mathcal{P}(I) \times \mathcal{P}(I)$ be the ancestor-descendant relationship. For any $X_1, X_2 \in I_G$, $X_1 \delta_{\mathcal{A}} X_2$ can be denoted when X_2 is obtained by replacing one or more items in X_1 with one of their descendants, X_1 is called an ancestor itemset of X_2 (and X_2 is called a descendant itemset of X_1).*

By using ancestor-descendant relationship, we can extend the original taxonomy (*1*-generalized itemset taxonomy) to express the ancestor-descendant relationships among *k*-length generalized itemsets.

Definition 4 (*k*-generalized itemset taxonomy) *The k-generalized itemset taxonomy is the partial order specified by an ancestor-descendant relationship $\delta_{\mathcal{A}}$ among generalized itemsets with the same k-length.*

Combining Two Relationships

The generalized itemsets can be shown in a complex view that combines both subset-superset and ancestor-descendant relationships. For example, assume the taxonomy as in Figure A.1C and a set of items $\{A, B, C, U, V\}$, the relationships among generalized itemsets are shown in Figure A.2. The solid lines express the subset-superset relationship where the lower itemset is a subset of the upper itemset, and the dashed lines express the ancestor-descendant relationship where the itemset at the beginning of an arrow is an ancestor itemset of the itemset at the end of the arrow.

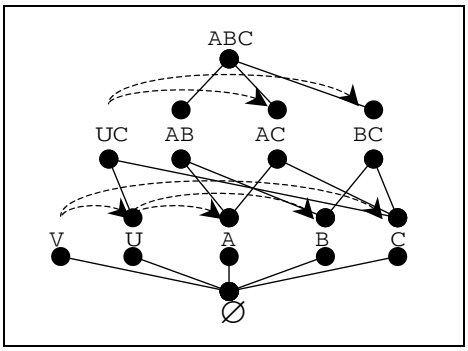


Figure A.2 Relationships on generalized itemsets (a part)

Constraints on Generalized Itemsets

We can exploit these two relationships as constraints for efficiently finding GFIs. Two lemmas are presented to justify the optimization as follows.

Lemma 1 (Subset-Superset Constraint) *For any $X \in I_G$, if a generalized itemset X is frequent, all subsets of X are frequent. Conversely, if a generalized itemset X is infrequent, all supersets of X are infrequent.*

Proof: Let $X, Y, Z \in I_G$ and $Z = X \cup Y$. The support of Z, $\sigma(Z) = |t(Z)| = |t(X) \cap t(Y)|$ must be less than or equal to the supports of its subsets, i.e. $\sigma(X)$ and $\sigma(Y)$. Thus, if Z satisfies minsup (frequent), both X and Y do too. If both or either of X and Y does not satisfy minsup (infrequent), then neither does Z.

For example, given minsup = 67%, a generalized itemset *ACD* ($\sigma(ACD)$=33%) is infrequent. The superset of *ACD*, such as *ACDE* ($\sigma(ACDE)$=33%) or *ABCDE* ($\sigma(ABCDE)$ = 17%), are also infrequent. This constraint shows that we need not consider the supersets of infrequent itemsets.

Lemma 2 (Ancestor-Descendant Constraints) *For any $X \in I_G$ where $\hat{X}$ is an ancestor itemset of X, if X is frequent, then $\hat{X}$ is also frequent. Conversely, if $\hat{X}$ is infrequent, X is also infrequent.*

Proof: Let $x, \hat{x} \in I$ and $Y, Z, \hat{Z} \subseteq I$. Assume that $Z = x \cup Y$, $\hat{Z} = \hat{x} \cup Y$. $\hat{x}$ is an ancestor of x, and $\hat{Z}$ is an ancestor itemset of Z. The support of $\hat{Z}$, $\sigma(\hat{Z}) = |t(\hat{Z})| = |t(\hat{x}) \cap t(Y)|$, must be greater than or equal to the support of Z, $\sigma(Z) = |t(Z)| = |t(x) \cap t(Y)|$, since $t(x) \subseteq t(\hat{x})$. Thus, if Z satisfies minsup (frequent), so does $\hat{Z}$. If $\hat{Z}$ does not satisfy minsup (infrequent), neither does Z.

For example, given minsup = 83%. A generalized itemset UE ($\sigma(UE) = 67\%$) is infrequent. The descendant itemsets of UE, such as AE ($\sigma(AE) = 33\%$) and BE ($\sigma(BE) = 33\%$), are also infrequent. This constraint shows that we need not consider the descendant itemsets of infrequent itemsets.

Generalized Closed Itemsets

In this section, the concept of generalized closed itemsets is defined by extending the traditional concept of closed itemsets in ARM [Pasquier et al., 1999, Zaki and Hsiao, 2002] to cope with the generalized itemsets. We also show that a set of generalized closed frequent itemsets is sufficient to be the representative of a larger set of GFIs. In order to understand the generalized closed frequent itemset, we introduce a maximal generalized itemset which is another representation of a generalized itemset.

Maximal Generalized Itemsets

In general, a generalized itemset can be transformed into another representation that includes both original items and all of their ancestors. This representation, we call a maximal generalized itemset of a generalized itemset. The formal definition is stated as follows.

Definition 5 (Maximal Generalized Itemset) *Let $X \subseteq I$, X is called a maximal generalized itemset iff the following condition is satisfied $\forall i(i \in X \rightarrow \mathcal{ANC}(i) \subset X)$.*

In every situation, each generalized itemset can always be transformed to each maximal generalized itemset and vice versa. Using the extended database, a generalized itemset can easily be transformed into the form of a maximal generalized itemset. This form is useful for finding generalized closed itemsets, since the concept of a closure finds a maximal superset of an itemset that supports the same tidset as a generalized itemset (described below). Thus, maximal generalized itemsets are generated instead of generalized itemsets when we find generalized closed itemsets. However each element in the set can be mapped to its corresponding generalized itemset.

Generalized Closed Itemset Concept

Definition 6 (Galois Connection) *Let $X \subseteq I$, and $Y \subseteq T$. Then the mapping functions,*
$t : \mathcal{P}(I) \mapsto \mathcal{P}(T), t(X) = \{y \in T | \forall x \in X, x\delta y\}$
$i : \mathcal{P}(T) \mapsto \mathcal{P}(I), i(Y) = \{x \in I | \forall y \in Y, x\delta y\}$
define a Galois connection between the power set of I and the power set of T.

The following properties hold for all $X, X_1, X_2 \subseteq I$ and $Y, Y_1, Y_2 \subseteq T$:

1. $X_1 \subseteq X_2 \rightarrow t(X_1) \supseteq t(X_2)$.
2. $Y_1 \subseteq Y_2 \rightarrow i(Y_1) \supseteq i(Y_2)$.
3. $X \subseteq i(t(X))$ and $Y \supseteq t(i(Y))$.

The mapping $t(X)$ is the maximal tidset which contains the generalized itemset X , given by $t(X) = \bigcap_{x \in X} t(x)$. The mapping $i(Y)$ is the maximal generalized itemset which is contained in the tidset Y, given by $i(Y) = \bigcap_{y \in Y} i(y)$. For example, $t(UDE) = t(U) \cap t(D) \cap$

$t(E) = 123456 \cap 13456 \cap 1356 = 1356$, and $i(356) = i(3) \cap i(5) \cap i(6) = VUBCWDE \cap VUABCWDE \cap VUBCWDE = VUBCWDE$.

Different from the original closure operator, the generalized closure operator is defined as follows:

Definition 7 (Generalized Closure) *Let $X \subseteq I$, and $Y \subseteq T$. The two mappings*

$$gc_{it} : \mathcal{P}(I) \mapsto \mathcal{P}(I),\ gc_{it}(X) = i \circ t(X) = i(t(X))$$
$$gc_{ti} : \mathcal{P}(T) \mapsto \mathcal{P}(T),\ gc_{ti}(Y) = t \circ i(Y) = t(i(Y))$$

are generalized closure operators on generalized itemset and tidset respectively. X is called a generalized closed itemset (GCI) when $X = gc_{it}(X)$, and Y is called a generalized closed tidset (GCT) when $Y = gc_{ti}(Y)$.

For $X \subseteq I$ and $Y \subseteq T$, the generalized closure operators gc_{it} and gc_{ti} satisfy the following properties:

1. $Y \subseteq gc_{ti}(Y)$.
2. $X \subseteq gc_{it}(X)$.
3. $gc_{it}(gc_{it}(X)) = gc_{it}(X)$, and $gc_{ti}(gc_{ti}(Y)) = gc_{ti}(Y)$.

The first property states that Y is a subset of its generalized closure. For example, let $Y = 135$, $gc_{ti}(135) = t(i(135)) = t(UCDE) = 1356$. Since $135 \neq gc_{ti}(135) = 1356$, such that 1356 is a generalized closed tidset while 135 is not. The second property says that X is a subset of its generalized closure. For example, $gc_{it}(VWDE) = i(t(VWDE) = i(1356) = VUCWDE$. Since $VWDE \neq gc_{it}(VWDE) = VUCWDE$, such that $VUCWDE$ is a GCI while $VWDE$ is not. Note that each GCI is a maximal generalized itemset, but it can be mapped to a generalized itemset. From the previous example, $VWDE$ and $VUCWDE$ can be transformed to the generalized itemsets VDE and $UCDE$, respectively. In generalized itemset form, this means that the GCI of VDE is $UCDE$. The last property says that the round-trip of mapping will obtain the same closure.

For any GCI X, there exists a companion GCT Y, with the property of $Y = t(X)$ and $X = i(Y)$. Such a GCI and GCT pair $X \times Y$ is called a *concept*. All possible concepts can form a Galois lattice of concepts as shown in Figure A.3.

Generalized Closed Frequent Itemsets

The support of a concept $X \times Y$ is a percentage of the size of closed tidset Y to the total number of transactions ($|Y|/|T|$). A GCI is called a *generalized closed frequent itemset* (GCFI) when its support is at least minsup.

Lemma 3 (Equivalence of Support) *For any generalized itemset X, its support is equal to the support of its generalized closure ($\sigma(X) = \sigma(gc_{it}(X))$).*

Proof: The support of X, $\sigma(X)$ is $|t(X)|/|T|$, and the support of $gc_{it}(X)$, $\sigma(gc_{it}(X))$ is $|t(gc_{it}(X))|/|T|$. To prove the lemma, we have to show that $t(X) = t(gc_{it}(X))$.

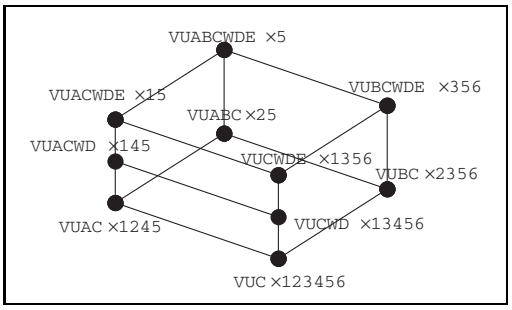


Figure A.3 Galois lattice of concepts and frequent concepts

Since gc_{ti} is a generalized closure operator, it satisfies the first property that $t(X) \subseteq gc_{ti}(t(X))$ $= t(i(t(X))) = t(gc_{it}(X))$. Thus $t(X) \subseteq t(gc_{it}(X))$. In the other case, the $gc_{it}(X)$ provides the maximal itemset, i.e., $X \subseteq gc_{it}(X)$, which implies that $t(X) \supseteq t(gc_{it}(X))$ due to the first property of Galois connection. Thus we conclude that $t(X) = t(gc_{it}(X))$.

Implicitly, the lemma states that all GFIs can be uniquely determined by the GCFIs since the support of any generalized itemsets will be equal to its generalized closure. Given a set of GCFIs, a Hasse diagram representing the subset-superset relationship among concepts in the Galois lattice, can be constructed using the method in [Valtchev et al., 2000] with $O(l.m.w(\mathcal{L}).d(\mathcal{L}))$ in time, where l is the average size of generalized itemsets, m is the number of items, $w(\mathcal{L})$ is the width of the lattice and $d(\mathcal{L})$ is the maximal degree of a lattice node. Consequently, all GFIs and their supports can be efficiently determined from the GCFIs and their Hasse diagram (Galois lattice). However, all GFIs need not be discovered, since a set of GCFIs is typically used to construct a minimal set of non-redundant rules as shown in [Bastide et al., 2000] and [Zaki, 2000]. In the worst case, the number of GCFIs is equal to the number of GFIs, but typically it is much smaller. From our example, there are 10 GCIs which are the representatives of a large number of all generalized itemsets as shown in Figure A.3. With minsup=50%, only 7 concepts (in bold font) are GCFIs.

Algorithms: *SET* and *cSET*

This section describes two algorithms, *SET* and *cSET*, that utilize two constraints for efficiently mining GFIs and GCFIs, respectively. For fast finding all GFIs, each of the lemma in section 4 can be applied to each relationship of generalized itemsets. Lemma 1 can be applied to the lattice of generalized itemsets while Lemma 2 can be applied to the taxonomies of *k*-generalized itemsets. Lemma 1 is concerned with the subset-superset relationship which exists in the generalized itemset lattice, while Lemma 2 is concerned with the ancestor-descendant relationship which exists in the taxonomies of *k*-generalized itemsets. These lemmas enable us to avoid generating itemsets that are dominantly infrequent. To enumerate all GFIs, we can traverse each relationship of generalized itemsets. For bi-directional traversal, the lattice of generalized itemsets should be traversed from subsets to their supersets and from ancestor itemsets to their descendant itemsets. Before generating any generalized itemsets, all of their subsets must be frequent. Similarly, an ancestor itemset must be frequent before generating its descendant itemsets. Following these approaches, only supersets and descendant itemsets of GFIs are generated.

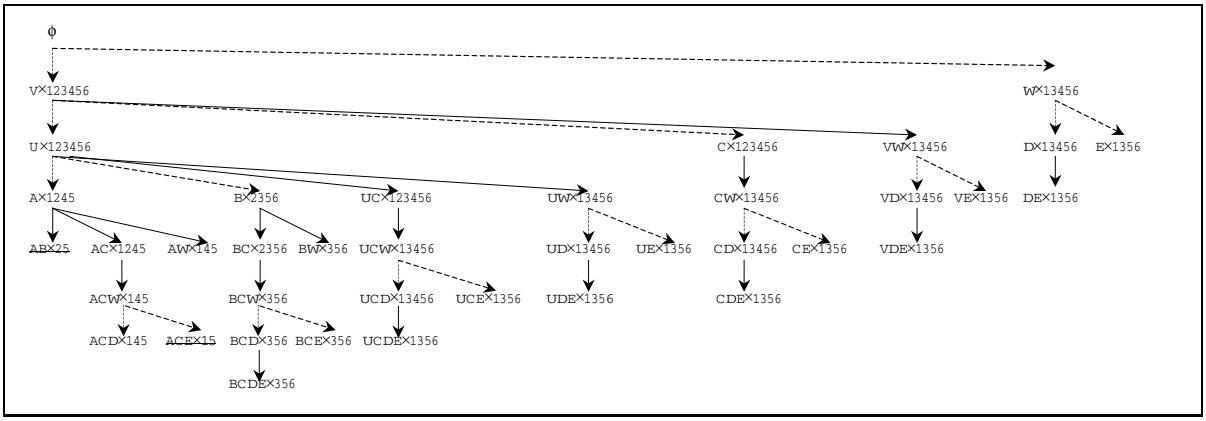


Figure A.4 Set enumeration using *SET* algorithm (minsup=50%)

SET Algorithm

Most of the computational cost on generating all GFIs is to count supports of the generalized itemsets for checking whether they are frequent or not, and checking to eliminate meaningless itemsets. The *SET* algorithm applies two techniques for enumerating GFIs using an extended vertical database format. The first one is to apply our novel set enumeration to generate only generalized itemsets without intensive checking on meaningless itemsets. This set enumeration was proposed in our previous work [Sriphaew and Theeramunkong, 2002]. The second technique is to apply a bi-directional traversal during set enumeration in order to avoid generating obvious infrequent itemsets.

As stated in section 5.1, a generalized itemset is transformed from a maximal generalized itemset by omitting the ancestors of items in the maximal set. However, the representation of a maximal generalized itemset is useful for describing the process of set enumeration. Normally, two itemsets can be joined together when they have the same size *k* and contain the preceding *k-1* itemset for avoiding redundant enumeration. Among maximal generalized itemsets, the join can be produced by a set union. For example, joining VUA with $VUC = VUAC$. However, when reducing to the generalized itemset, the join can be produced by a set union of the first itemset with the last item of the second itemset, for example, joining A with $UC = A \cup C = AC$ where its tidset is given by a set intersection. This join operation on generalized itemsets is used in the *SET* algorithm.

For clarity, we explain the *SET* algorithm by the example illustrated in Figure A.4. With minsup=50%, the proposed set enumeration starts with an empty set. Then, all second-leveled items of the taxonomy which are frequents, i.e. V and W, are added to the second level of the tree. The children under each generalized itemset are generated in two manners. First, we generate *taxonomy-based child itemsets* (based on ancestor-descendant relationship) by replacing the right most item of that generalized itemset with one of their children (if any). Secondly, we generate all *join-based child itemsets* (based on subset-superset relationship) by the union between the generalized itemsets and the last item of their siblings that have higher orders. For example, generating the children of itemset V, we first generate taxonomy-based child itemsets, i.e., U and C, and then generate join-based child itemsets, i.e., VW. Each generalized itemset which is generated must be frequent, otherwise it will be pruned. With the same approach, the process recursively occurs until no new GFI is

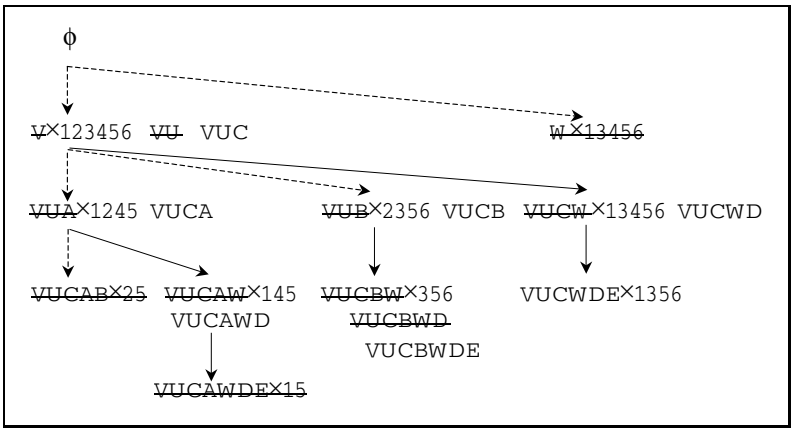


Figure A.5 Set enumeration using *cSET* algorithm (minsup=50%)

generated. Finally, a complete GFI tree is constructed without excessive checking cost. All generalized itemsets in Figure A.4, except ones with a cross, are GFIs. The pseudo-codes of *SET* will be shown in Section 6.3.

cSET Algorithm

This section presents an extension of the *SET* algorithm, called the *cSET* algorithm, for mining GCFIs. Since the GCFI is in the form of the maximal generalized itemset, we intend to enumerate the generalized itemsets in the form of maximal generalized itemsets. The same process of set enumeration in *SET* for bi-directional traversal is used, but the join operation is given by a set union on maximal generalized itemsets with some conditional properties to discard non-GCFIs.

In the process of set enumeration, the following conditional properties are used to reduce the number of GCFIs that need to be generated. Assume that $X_1 \times t(X_1)$ is joined with $X_2 \times t(X_2)$:

1. If $t(X_1) = t(X_2)$, then (1) replace every occurrence of X_1 with $X_1 \cup X_2$, (2) remove X_2 if X_2 is a sibling of X_1, and (3) generate taxonomy-based child itemsets of the current new X_1 (since the former X_1 is replaced by $X_1 \cup X_2$).

2. If $t(X_1) \subset t(X_2)$, then (1) replace every occurrence of X_1 with $X_1 \cup X_2$, and (2) generate taxonomy-based child itemsets of the current new X_1.

3. If $t(X_1) \supset t(X_2)$ and $t(X_1) \cap t(X_2)$ is not contained in hash table, then (1) store $t(X_1) \cap t(X_2)$ in the hash table, (2) remove X_2 if X_2 is a sibling of X_1, and (3) generate $X_1 \cup X_2$ under X_1 in tree.

4. If $t(X_1) \neq t(X_2)$ and $t(X_1) \cap t(X_2)$ is not contained in hash table, (1) store $t(X_1) \cap t(X_2)$ in the hash table, and (2) generate $X_1 \cup X_2$ under X_1 in tree.

For clarity, we explain the *cSET* algorithm using the example in Figure A.5. With minsup=50%, the *cSET* algorithm starts with an empty set. Then, all second-leveled items of the

taxonomy which are frequent, i.e., V and W, are added to the second level of tree. Similar to *SET*, children are generated based on two methods but the form of an itemset has changed to be the maximal generalized itemset. The taxonomy-based child itemset is generated by a set union between the current itemset and one of the children of the rightmost item in that set according to taxonomy (if any). The join-based child itemset is normally generated by a set union on maximal generalized itemsets as previously described in section 6.1. The first child of V from taxonomy (i.e., U) produces taxonomy-based itemset VU. The first property holds for VU, which results in replacing V with VU and then generating its taxonomy-based itemsets, i.e., VUA and VUB with the fourth property. The second child of V from taxonomy (i.e., C) is still joined with the current itemset (VU), which produces VUC. Again, the first property holds for VUC, which results in replacing VU and the children in the tree under VU with VUC. Because of this, VUA and VUB are replaced by $VUCA$ and $VUCB$, respectively. Next, the current itemset (VUC) joins with its sibling (W), i.e., $VUCW$. The third property holds for $VUCW$, which results in removing W and then generating $VUCW$ under the current VUC. This process shows that the first and third properties can help us to discard some itemsets and the subtrees under those itemsets which are clearly not to be GCFIs. That is, W and the subtree under W are pruned. With the same approach, the process recursively occurs until no new GCFI is generated. The hash table is used for checking whether the current tidset occurs in the previous enumeration or not. Instead of 36 GFIs in Figure A.4, we enumerate only 7 GCFIs as shown in Figure A.5. This action results in reducing computational time. All remaining maximal generalized itemsets in Figure A.5, except ones with a cross, are GCFIs.

Pseudo-Codes Description

The pseudo-codes of *SET* and its extension *cSET* are shown in Figure A.6. For the *SET* algorithm, line 11.c, 13.c, 17.c and 18.c-30.c are ignored. In the main procedure *SET-Main*, the subordinate function called *SET-Extend*, recursively creates a subtree using the proposed method. The *GenTaxChild* function produces a taxonomy-based child itemset while the *GenJoinChild* function produces a join-based child itemset. In line 12 and 16, the generated itemsets must be checked to ensure whether they are frequent or not. The *NewNode* function generates a new itemset under the current itemset. For the *cSET* algorithm, line 11.s, 13.s and 17.s are ignored. Similar to *SET*, *cSET* uses *SET-Main*, *SET-Extend*, *GenTaxChild* and *GenJoinChild* as well as *cSET-Property* and *Hash* functions. Since the form of a generalized itemset in *SET* is generalized itemset but *cSET* is maximal generalized itemset, line 13 of *SET* and *cSET* are different. Instead of *NewNode* function in line 13.s and 17.s, we use the *cSET-Property* function in line 13.c and 17.c to check the conditional properties as previously described in section 6.2 for generating only GCFIs. The *Hash* function is used for checking whether the current tidset occurs in the previous enumeration or not by returning 1 when it exists, or storing that tidset in the hash table and return 0 when it does not exist. Following the *SET* algorithm, we will get the tree of all GFIs while the *cSET* algorithm will get the tree of all GCFIs.

Experimental Results

For testing the performance of our approaches, we compare the *SET* and *cSET* algorithms with two popular algorithms, i.e., Cumulate [Srikant and Agrawal, 1997] and Prutax [Hipp et al., 1998]. All algorithms were coded in C. Experiments were done on a 1.7GHz Pentium IV PC with 640MB of main memory, running Windows 2000.

To measure the exact execution time of algorithms (excluding intensive I/O cost), we make the dataset and its taxonomy reside in the memory. Therefore, the memory size should be large enough to store the data in order to avoid page swapping time. Here, we can illustrate the calculation of memory needed for the largest dataset as follows. The largest dataset in the experiments is the synthetic dataset with 1 million (10^6) transactions which contains 10 items per transaction and 5 fanouts per item. Converting the dataset to the vertical format, where an item is encoded to an integer, the required memory for this dataset is $10^6 \times 10 \times 5 \times 4$ bytes $\approx$ 200 MB. From our investigation, the exact memory usage for this dataset including other variables of the program and required memory for the operating system is at least 32 MB. Approximately, 300 MB are required where the remaining memory space can be used for the computation process. To process more transactions, we need more memory. Anyway, 640 MB is large enough for the current experiments.

Table A.1 The default value of parameters in synthetic datasets

Parameter	Default
Number of transactions	100K
Average size of the transaction	10
Number of items	100K
Number of roots	250
Fanout	5
Depth-ratio	1
Minimum support	1%

$$\text{Depth-ratio} = \frac{\text{probability that item in a rule comes from level i}}{\text{probability that item comes from level i+1}}$$

Table A.2 The real datasets and their parameters

Dataset	Parameters			
	#Trans	#Items*	#Roots	Fanout
MushroomR40F3	8124	159	40	3
MushroomR24F5	8124	143	24	5
ChessR15F5	3196	90	15	5

*#Items include both leaf (original) items and non-leaf items.

Datasets

The synthetic and real datasets are used as benchmarks for evaluating the performance. The synthetic datasets were automatically generated by a generator tool[1]. They mimic the transactions in a retailing environment. The important default values of parameters in synthetic datasets are shown in Table A.1. Two standard real datasets, i.e., mushroom and chess[2], are also used for investigating our methods in an actual environment. These real datasets are often used for testing the performance of data mining algorithms. There is no taxonomy specified in the original real datasets. Therefore, we construct an additional taxonomy for each dataset by defining a number of roots and fanout of the taxonomy in order to make all original items appear in the leaf level of the taxonomy. All 119 original items of mushroom can be covered in the second depth of a taxonomy with the number of roots and fanout, respectively, being 40 and 3 (or 24 and 5), and 75 original items of chess can be covered in the second depth of taxonomy with the number of roots and fanout, respectively, being 15 and 5. These taxonomies are suitable since the original real datasets we used are dense and many items usually appear in most portions of transactions. Therefore, with higher fanout the excessive amount of dense patterns may occur and then the algorithms suffer from the memory limitation problem. Thus, we get three different datasets as shown in Table A.2. These three real datasets are fixed throughout the experiments.

Performance Testing

Four experiments are performed to investigate the performance of the algorithms in different situations. At first, we study how the algorithms perform on different characteristics of taxonomy. Secondly, we investigate the performances of the algorithms on different scaling of database. Thirdly, the performance of algorithms with various minsups is evaluated, and the numbers of frequent patterns (i.e., GFIs and GCFIs) are compared. Finally, the memory usage of each algorithm is checked using both synthetic and real datasets. We only use real datasets in the third and fourth experiments since it is not possible to vary the characteristics of their taxonomy.

Taxonomy Characteristics: Figure A.7 shows the execution time of algorithms when the characteristics of taxonomy are changed. The performances of *SET* and *cSET* are so close since the numbers of GFIs and GFCIs almost equal (shown in the latter experiment). Both algorithms are approximately 4 to 180 times faster than Prutax and 22 to 230 times faster than Cumulate. In case of the smaller number of roots, taxonomy levels become deeper and then the number of ancestor itemsets turns out to be larger. *SET* and *cSET* are not sensitive to this situation, while Prutax requires more time for checking and Cumulate needs more time to modify the transactions. With different fanouts, the number of children of each non-leaf item in taxonomy is varied. The number of ancestor itemsets in lower fanouts is larger than higher fanouts. As shown in the figure, decreasing the fanout has an effect similar to decreasing the number of roots. For a lower depth ratio, we gain more frequent patterns that contain items coming from the lower parts rather than the upper parts of taxonomy. The number of ancestor itemsets increases and this phenomenon results in more time consumed

[1]The generator tool are provided by IBM Almaden Site.

[2]Original mushroom and chess are provided by UCI Machine Learning Database Repository.

in Prutax and Cumulate. *SET* and *cSET* are approximately 6-10 times faster than Prutax and 20-38 times faster than Cumulate with depth-ratio variation.

Scaling Database: Figure A.8 shows the execution time of each algorithm when the database is scaled up and down. In this experiment, all taxonomy parameters are fixed to their default values, but only the number of transactions and the number of items are scaled. We observe an exponential increment in the running time with the increasing number of transactions. However, *SET* and *cSET* still perform well with the large number of transactions. With the scaling number of items, *SET* and *cSET* are not affected by this variation since an item occurs sparsely in the transactions and then the number of GFIs to be counted is reduced, but it results in more time consumed in Prutax and Cumulate.

Table A.3: Experimental results: minimum support variation and number of frequent patterns

Minsup	Execution Time (sec)				#Frequent Patterns	
	SET	*cSET*	Prutax	Cumulate	#GFIs	#GFCIs
Dataset: SynR250F5D1						
4	0.7	0.7	9.5	30.6	228	226
3	1.0	1.0	11.3	37.7	404	401
2	1.5	1.4	14.9	58.5	848	843
1.5	2.0	1.8	19.3	73.2	1,484	1,475
1	2.9	2.6	29.9	101.1	3,235	3,211
0.75	4.0	3.3	40.8	71563.6	5,684	5,633
Dataset: MushroomR40F3						
100	0.03	0.01	0.03	0.39	95	1
90	0.06	0.02	0.06	0.69	431	5
80	0.31	0.02	0.38	1.58	1,839	18
70	1.09	0.05	1.16	3.19	7,983	42
60	2.92	0.05	2.98	8.59	19,543	102
50	9.48	0.13	9.69	57.09	68,095	297
Dataset: MushroomR24F5						
100	0.14	0.01	0.05	0.52	191	1
90	1.41	0.03	1.23	3.60	9,023	34
80	4.59	0.06	4.00	14.19	28,127	84
70	12.88	0.13	12.27	79.77	85,343	216
60	29.94	0.28	29.14	360.77	204,143	485
50	76.67	0.59	78.36	2141.99	524,231	1,102
Dataset: ChessR15F5						
100	2.36	0.01	2.64	11.08	32,767	1
98	15.41	0.01	20.03	401	231,423	22
96	25.45	0.02	158.14	1127.92	389,119	45
94	65.69	0.02	926.25	7081.02	935,935	125
92	110.75	0.05	2895.33	19334.05	1,602,559	270
90	179.53	0.06	7120.33	48890.14	2,565,631	499

Minsup Variation and Number of Frequent Patterns: Typically, the real datasets are very dense, i.e., frequent patterns are mostly long even high values of minsup while the synthetic

datasets are sparse. Table A.3 shows the execution time of each algorithm and the number of frequent patterns when minsup is varied. The execution time shows exponential growth with decreasing minsup. *SET* and *cSET* provide similar performance in the synthetic dataset (SYNR250F5D1), but they perform differently on the real datasets (i.e., MushroomR40F3, MushroomR24F5 and ChessR15F5). Cumulate cannot be executed with a lower minsup in a synthetic dataset since it generates a lot of candidates which are at last infrequent. In the real datasets, the performances of *SET* and Prutax are quite close since the sizes of real datasets are small, resulting in a trivial hashing time for Prutax. However, *cSET* still performs better than other algorithms, since the number of GCFIs is significantly smaller than the number of GFIs as shown in Table A.3. We observe that the difference between the number of GFIs and GCFIs is much smaller in the synthetic datasets but dominantly larger in the real datasets.

Table A.4 Maximum memory usage of each algorithms

Dataset(%Minsup)	Maximum Memory Usage (MB)			
	SET	*cSET*	Prutax	Cumulate
SynR250F5D1(2%)	35.9	44.6	52.8	28.2
SynR250F5D1(1%)	46.2	48.9	55.0	136.3
MushroomR40F3(80%)	12.9	11.6	16.3	23.4
MushroomR40F3(60%)	13.2	13.7	17.4	29.8
MushroomR24F5(80%)	13.0	13.4	16.4	29.4
MushroomR24F5(60%)	13.8	22.3	17.8	79.9
ChessR15F5(100%)	10.5	9.4	13.9	35.9
ChessR15F5(98%)	10.8	9.8	14.9	72.4
ChessR15F5(96%)	10.9	10.4	15.2	138.4

Memory Usage: Table A.4 shows the maximum memory usage of each algorithm with different minsups in the synthetic and real datasets. For the synthetic dataset, the memory usage of *cSET* is slightly greater than *SET* since the number of their frequent patterns are almost equal and *cSET* has to hold some GFIs in memory for checking. However, their memory usage are rather smaller than Prutax. For the real datasets, the memory usage of *SET* and *cSET* are smaller than the other two algorithms, but the memory usage of Cumulate grows excessively since a lot of candidates are generated and held in memory. These results confirm that *SET* and *cSET* are superior to the other algorithms in memory utilization.

Summary

In this work, we presented a theoretical framework of generalized itemsets based on subset-superset relationship (represented by lattice of generalized itemsets), and ancestor-descendant relationship (represented by taxonomy of k-generalized itemsets). To efficiently discover all generalized frequent itemsets, we introduced two constraints corresponding to these two relationships. We proposed *SET* and *cSET* algorithms to enumerate generalized frequent itemsets and generalized closed frequent itemsets, respectively. *SET* and *cSET* use an efficient traversal on the combination of two relationships to avoid generating meaningless itemsets, and apply two constraints to prevent counting useless generalized itemsets that are obviously

infrequent. This lets *SET* and *cSET* efficiently find all frequent patterns. A number of experiments showed that *SET* and *cSET* outperform the previous well-known algorithms in both computational time and memory utilization, especially for real situations. There are other problems related to ARM to be considered in GARM, including incremental data mining, constraint-based mining, interesting measures, negative rule mining, parallel mining, and so on. They are left as our further explorations.

```
SET-Main( D, T, minsup)
01      Root = Null Tree;
02      For each x in Second-level items of T
03        If ||t(x)|| ≥ minsup then NewNode(Root, x);
04      SET − Extend(Root);

SET-Extend(Node)        //Recursively generate tree in depth-first
05      For each F_i ∈ Node.Child in T {
06        F_i.Child = NULL;
07        GenTaxChild(F_i);
08        GenJoinChild(F_i);
09        If F_i.Child ≠ NULL then SET − Extend(F_i);}

GenTaxChild(F_i)        //Generate taxonomy-based child itemsets
10      For each x ∈ LastItem(F_i).Child in T {
11.s      C = F_i after replace LastItem(F_i) with x;      //For SET
11.c      C = F_i ∪ x;                                     //For cSET
12        If |t(C)| ≥ minsup then {
13.s        NewNode(F_i, C);                               //For SET
13.c        cSET.Property(F_i, x, C);}                     //For cSET

GenJoinChild(F_i)       //Generate join-based child itemsets
14      For each F_j ∈ F_i.Sibling{       // j > i
15        C = F_i ∪ LastItem(F_j);
16        If |t(C)| ≥ minsup then {
17.s        NewNode(F_i, C);                             //For SET
17.c        cSET.Property(F_i, F_j, C);}                 //For cSET

cSET.Property(F_i, F_j, C)      //4 Properties of cSET
18.c    If t(F_i) = t(F_j) then {                        //Prop.1
19.c      Replace all F_i with C;
20.c      if F_j ∈ F_i.sibling then Remove(F_j);
21.c      GenTaxChild(F_i);}
22.c    else if t(F_i) ⊂ t(F_j) then {                   //Prop.2
23.c      Replace all F_i with C;
24.c      GenTaxChild(F_i);}
25.c    else if t(F_i) ⊃ t(F_j) then {                   //Prop.3
26.c      If F_j ∈ F_i.sibling then Remove(F_j);
27.c      If !Hash(t(C)) then NewNode(F_i, C);}
28.c    else if !Hash(t(C)) then NewNode(F_i, C);        //Prop.4

Hash(tidset)        //Find tidset in Hash Table
29.c    if Found tidset in Hash Table then return 1;
30.c    else Add tidset in Hash Table; return 0;
```

Figure A.6 The pseudo-codes of *SET* and *cSET* algorithm

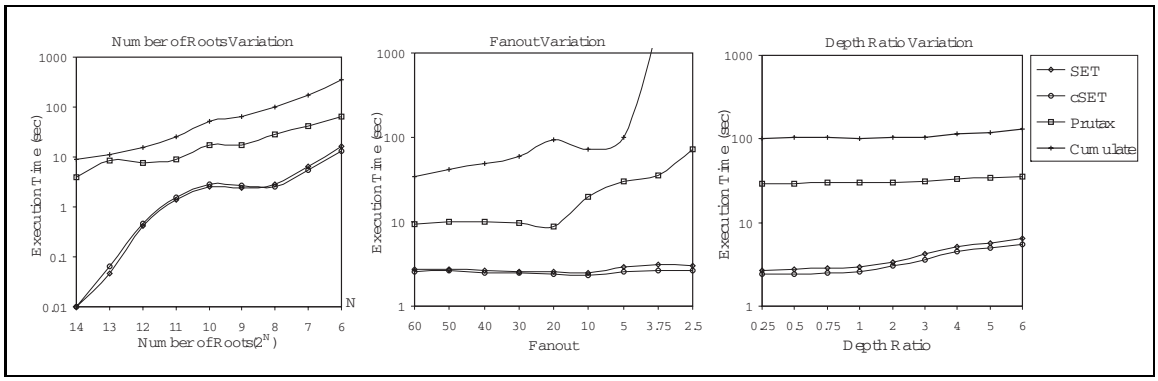


Figure A.7 Experimental results: taxonomy characteristics

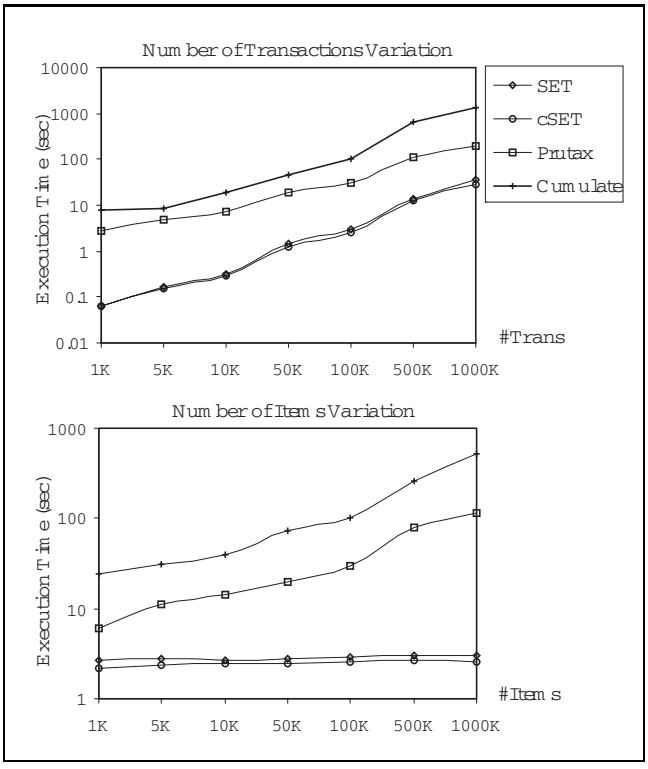


Figure A.8 Experimental results: scaling database

Appendix B

Stoplist and Stemming Algorithm

Porter Stemming Algorithm [Porter, 1980]

To present the suffix stripping algorithm in its entirety we will need a few definitions.

A *consonant* in a word is a letter other than A, E, I, O or U, and other than Y preceded by a consonant. (The fact that the term 'consonant' is defined to some extent in terms of itself does not make it ambiguous.) So in TOY the consonants are T and Y, and in SYZYGY they are S, Z and G. If a letter is not a consonant it is a *vowel*.

A consonant will be denoted by c, a vowel by v. A list ccc... of length greater than 0 will be denoted by C, and a list vvv... of length greater than 0 will be denoted by V. Any word, or part of a word, therefore has one of the four forms:

```
CVCV ... C
CVCV ... V
VCVC ... C
VCVC ... V
```

These may all be represented by the single form

```
[C]VCVC ... [V]
```

where the square brackets denote arbitrary presence of their contents. Using (VC)m to denote VC repeated m times, this may again be written as

```
[C](VC){m}[V].
```

m will be called the *measure* of any word or word part when represented in this form. The case m = 0 covers the null word. Here are some examples:

```
m=0    TR,  EE,  TREE,  Y,  BY.
m=1    TROUBLE,  OATS,  TREES,  IVY.
m=2    TROUBLES,  PRIVATE,  OATEN,  ORRERY.
```

The *rules* for removing a suffix will be given in the form

```
(condition) S1 -> S2
```

This means that if a word ends with the suffix S1, and the stem before S1 satisfies the given condition, S1 is replaced by S2. The condition is usually given in terms of m, e.g.

```
(m > 1) EMENT ->
```

Here S1 is 'EMENT' and S2 is null. This would map REPLACEMENT to REPLAC, since REPLAC is a word part for which m = 2.

The 'condition' part may also contain the following: *S - the stem ends with S (and similarly for the other letters). *v* - the stem contains a vowel. *d - the stem ends with a double consonant (e.g. -TT, -SS). *o - the stem ends cvc, where the second c is not W, X or Y (e.g. -WIL, -HOP).

And the condition part may also contain expressions with *and*, *or* and *not*, so that

```
(m>1 and (*S or *T))
```

tests for a stem with m>1 ending in S or T, while

```
(*d and not (*L or *S or *Z))
```

tests for a stem ending with a double consonant other than L, S or Z. Elaborate conditions like this are required only rarely.

In a set of rules written beneath each other, only one is obeyed, and this will be the one with the longest matching S1 for the given word. For example, with

```
SSES -> SS
IES  -> I
SS   -> SS
S    ->
```

(here the conditions are all null) CARESSES maps to CARESS since SSES is the longest match for S1. Equally CARESS maps to CARESS (S1='SS') and CARES to CARE (S1='S').

For the rules below, examples of their application, successful or otherwise, are given on the right in lower case. The algorithm now follows:

Step 1a

```
SSES -> SS                      caresses  ->  caress
IES  -> I                       ponies    ->  poni
                                ties      ->  ti
SS   -> SS                      caress    ->  caress
S    ->                         cats      ->  cat
```

Step 1b

```
(m>0) EED -> EE                    feed      ->  feed
                                   agreed    ->  agree
(*v*) ED  ->                       plastered ->  plaster
                                   bled      ->  bled
(*v*) ING ->                       motoring  ->  motor
                                   sing      ->  sing
```

If the second or third of the rules in Step 1b is successful, the following is done:

```
AT -> ATE                      conflat(ed)  ->  conflate
BL -> BLE                      troubl(ed)   ->  trouble
IZ -> IZE                      siz(ed)      ->  size
(*d and not (*L or *S or *Z))
   -> single letter
                               hopp(ing)    ->  hop
                               tann(ed)     ->  tan
                               fall(ing)    ->  fall
                               hiss(ing)    ->  hiss
                               fizz(ed)     ->  fizz
(m=1 and *o) -> E              fail(ing)    ->  fail
                               fil(ing)     ->  file
```

The rule to map to a single letter causes the removal of one of the double letter pair. The -E is put back on -AT, -BL and -IZ, so that the suffixes -ATE, -BLE and -IZE can be recognised later. This E may be removed in step 4.

Step 1c

```
(*v*) Y -> I                   happy        ->  happi
                               sky          ->  sky
```

Step 1 deals with plurals and past participles. The subsequent steps are much more straightforward.

Step 2

```
(m>0) ATIONAL ->  ATE          relational     ->  relate
(m>0) TIONAL  ->  TION         conditional    ->  condition
                               rational       ->  rational
(m>0) ENCI    ->  ENCE         valenci        ->  valence
(m>0) ANCI    ->  ANCE         hesitanci      ->  hesitance
(m>0) IZER    ->  IZE          digitizer      ->  digitize
(m>0) ABLI    ->  ABLE         conformabli    ->  conformable
(m>0) ALLI    ->  AL           radicalli      ->  radical
(m>0) ENTLI   ->  ENT          differentli    ->  different
(m>0) ELI     ->  E            vileli        - >  vile
(m>0) OUSLI   ->  OUS          analogousli    ->  analogous
(m>0) IZATION ->  IZE          vietnamization ->  vietnamize
```

```
(m>0) ATION   ->  ATE          predication    ->  predicate
(m>0) ATOR    ->  ATE          operator       ->  operate
(m>0) ALISM   ->  AL           feudalism      ->  feudal
(m>0) IVENESS ->  IVE          decisiveness   ->  decisive
(m>0) FULNESS ->  FUL          hopefulness    ->  hopeful
(m>0) OUSNESS ->  OUS          callousness    ->  callous
(m>0) ALITI   ->  AL           formaliti      ->  formal
(m>0) IVITI   ->  IVE          sensitiviti    ->  sensitive
(m>0) BILITI  ->  BLE          sensibiliti    ->  sensible
```

To fasten the test for the string S1, doing a program switch on the penultimate letter of the word being tested is applied. This gives a fairly even breakdown of the possible values of the string S1. It will be seen in fact that the S1-strings in step 2 are presented here in the alphabetical order of their penultimate letter. Similar techniques may be applied in the other steps.

Step 3

```
(m>0) ICATE ->  IC             triplicate     ->  triplic
(m>0) ATIVE ->                 formative      ->  form
(m>0) ALIZE ->  AL             formalize      ->  formal
(m>0) ICITI ->  IC             electriciti    ->  electric
(m>0) ICAL  ->  IC             electrical     ->  electric
(m>0) FUL   ->                 hopeful        ->  hope
(m>0) NESS  ->                 goodness       ->  good
```

Step 4

```
(m>1) AL    ->                 revival        ->  reviv
(m>1) ANCE  ->                 allowance      ->  allow
(m>1) ENCE  ->                 inference      ->  infer
(m>1) ER    ->                 airliner       ->  airlin
(m>1) IC    ->                 gyroscopic     ->  gyroscop
(m>1) ABLE  ->                 adjustable     ->  adjust
(m>1) IBLE  ->                 defensible     ->  defens
(m>1) ANT   ->                 irritant       ->  irrit
(m>1) EMENT ->                 replacement    ->  replac
(m>1) MENT  ->                 adjustment     ->  adjust
(m>1) ENT   ->                 dependent      ->  depend
(m>1 and (*S or *T)) ION ->    adoption       ->  adopt
(m>1) OU    ->                 homologou      ->  homolog
(m>1) ISM   ->                 communism      ->  commun
(m>1) ATE   ->                 activate       ->  activ
(m>1) ITI   ->                 angulariti     ->  angular
(m>1) OUS   ->                 homologous     ->  homolog
(m>1) IVE   ->                 effective      ->  effect
(m>1) IZE   ->                 bowdlerize     ->  bowdler
```

The suffixes are now removed. All that remains is a little tidying up.

Step 5a

```
(m>1) E     ->                  probate        ->  probat
                                rate           ->  rate
(m=1 and not *o) E ->           cease          ->  ceas
```

Step 5b

```
(m > 1 and *d and *L) -> single letter
                                controll       ->  control
                                roll           ->  roll
```

The algorithm is careful not to remove a suffix when the stem is too short, the length of the stem being given by its measure, m. There is no linguistic basis for this approach. It was merely observed that m could be used quite effectively to help decide whether or not it was wise to take off a suffix. For example, in the following two lists:

```
list A        list B
------        ------
RELATE        DERIVATE
PROBATE       ACTIVATE
CONFLATE      DEMONSTRATE
PIRATE        NECESSITATE
PRELATE       RENOVATE
```

-ATE is removed from the list B words, but not from the list A words. This means that the pairs DERIVATE/DERIVE, ACTIVATE/ACTIVE, DEMONSTRATE/DEMONS- TRABLE, NECESSITATE/NECESSITOUS, will conflate together. The fact that no attempt is made to identify prefixes can make the results look rather inconsistent. Thus PRELATE does not lose the -ATE, but ARCHPRELATE becomes ARCHPREL. In practice this does not matter too much, because the presence of the prefix decreases the probability of an erroneous conflation.

Complex suffixes are removed bit by bit in the different steps. Thus GENERALIZATIONS is stripped to GENERALIZATION (Step 1), then to GENERALIZE (Step 2), then to GENERAL (Step 3), and then to GENER (Step 4). OSCILLATORS is stripped to OSCILLATOR (Step 1), then to OSCILLATE (Step 2), then to OSCILL (Step 4), and then to OSCIL (Step 5).

List of 524 English Stopwords from SMART System [Rocchio, 1971]

Table B.1 List of 524 English stopwords

a	able	about	above	according	accordingly
across	actually	after	afterwards	again	against
all	allow	allows	almost	alone	along
already	also	although	always	am	among
amongst	an	and	another	any	anybody
anyhow	anyone	anything	anyway	anyways	anywhere
apart	appear	appreciate	appropriate	are	around
as	aside	ask	asking	associated	at
available	away	awfully	b	be	became
because	become	becomes	becoming	been	before
beforehand	behind	being	believe	below	beside
besides	best	better	between	beyond	both
brief	but	by	c	came	can
cannot	cant	cause	causes	certain	certainly
changes	clearly	co	com	come	comes
concerning	consequently	consider	considering	contain	containing
contains	corresponding	could	course	currently	d
definitely	described	despite	did	different	do
does	doing	done	down	downwards	during
e	each	edu	eg	eight	either
else	elsewhere	enough	entirely	especially	et
etc	even	ever	every	everybody	everyone
everything	everywhere	ex	exactly	example	except
f	far	few	fifth	first	five
followed	following	follows	for	former	formerly
forth	four	from	further	furthermore	g
get	gets	getting	given	gives	go
goes	going	gone	got	gotten	greetings
h	had	happens	hardly	has	have
having	he	hello	help	hence	her
here	hereafter	hereby	herein	hereupon	hers
herself	hi	him	himself	his	hither
hopefully	how	howbeit	however	i	ie
if	ignored	immediate	in	inasmuch	inc
indeed	indicate	indicated	indicates	inner	insofar
instead	into	inward	is	it	its
itself	j	just	k	keep	keeps
kept	know	knows	known	l	last
lately	later	latter	latterly	least	less
lest	let	like	liked	likely	little
look	looking	looks	ltd	m	mainly
many	may	maybe	me	mean	meanwhile
merely	might	more	moreover	most	mostly
much	must	my	myself	n	name

Table B.2 List of 524 English stopwords (continue)

namely	nd	near	nearly	necessary	need
needs	neither	never	nevertheless	new	next
nine	no	nobody	non	none	noone
nor	normally	not	nothing	novel	now
nowhere	o	obviously	of	off	often
oh	ok	okay	old	on	once
one	ones	only	onto	or	other
others	otherwise	ought	our	ours	ourselves
out	outside	over	overall	own	p
particular	particularly	per	perhaps	placed	please
plus	possible	presumably	probably	provides	q
que	quite	qv	r	rather	rd
re	really	reasonably	regarding	regardless	regards
relatively	respectively	right	s	said	same
saw	say	saying	says	second	secondly
see	seeing	seem	seemed	seeming	seems
seen	self	selves	sensible	sent	serious
seriously	seven	several	shall	she	should
since	six	so	some	somebody	somehow
someone	something	sometime	sometimes	somewhat	somewhere
soon	sorry	specified	specify	specifying	still
sub	such	sup	sure	t	take
taken	tell	tends	th	than	thank
thanks	thanx	that	thats	the	their
theirs	them	themselves	then	thence	there
thereafter	thereby	therefore	therein	theres	thereupon
these	they	think	third	this	thorough
thoroughly	those	though	three	through	throughout
thru	thus	to	together	too	took
toward	towards	tried	tries	truly	try
trying	twice	two	u	un	under
unfortunately	unless	unlikely	until	unto	up
upon	us	use	used	useful	uses
using	usually	uucp	v	value	various
very	via	viz	vs	w	want
wants	was	way	we	welcome	well
went	were	what	whatever	when	whence
whenever	where	whereafter	whereas	whereby	wherein
whereupon	wherever	whether	which	while	whither
who	whoever	whole	whom	whose	why
will	willing	wish	with	within	without
wonder	would	would	x	y	yes
yet	you	your	yours	yourself	yourselves
z	zero				

Appendix C

Some Examples of Publications and Their References

This section present some examples of publications and their references that are used in the experiments. Only references that exist in our collections are shown here. However, the document relations among citer (cite from) and citee (cite to) publications including the relations among citee publications of one publication can be explicitly shown by the title of publications. Three examples are shown below.

Example 1: K.-T. Cheng, Gate-level test generation for sequential circuits. ACM Transactions on Design Automation of Electronic Systems, 1(4):405442, 1996.

References (a part):

- J.A. Abraham and V.K. Agarwal, Test generation for digital systems, Fault-tolerant computing: theory and techniques; vol. 1, Prentice-Hall, Inc., Upper Saddle River, NJ, 1986.
- S.T. Chakradhar and S.G. Rothweiler, Redundancy Removal and Test Generation for Circuits with Non-Boolean Primitives, Proceedings of the 13th IEEE VLSI Test Symposium (VTS'95), p.12, April 30-May 03, 1995.
- K.-T. Cheng and V.D. Agrawal, Unified Methods for VLSI Simulation and Test Generation, Kluwer Academic Publishers, Norwell, MA, 1989.
- F. Corno, P. Prinetto, M. Rebaudengo, M. Sonza Reorda and R. Mosca, Advanced Techniques for GA-based sequential ATPGs, Proceedings of the 1996 European conference on Design and Test, p.375, March 11-14, 1996.
- S. Devadas, H-K.Y. Ma and A.R. Newton, Redundancies and don't cares in sequential logic synthesis, Journal of Electronic Testing: Theory and Applications, v.1 n.1, p.15-30, Feb. 1990.
- A. Ghosh, S. Devadas and A.R. Newton, Sequential test generation at the register-transfer and logic levels, Proceedings of the 27th ACM/IEEE conference on Design automation, p.580-586, June 24-27, 1990, Orlando, Florida, United States.
- U. Gl*ä*sser and H.T. Vierhaus, FOGBUSTER: an efficient algorithm for sequential test generation, Proceedings of the conference on European design automation, p.230-235, September 18-22, 1995, Brighton, England.
- D.E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, 1989.
- M.C. Hansen and J.P. Hayes, High-Level Test Generation Using Symbolic Scheduling, Proceedings of the IEEE International Test Conference on Driving Down the Cost of Test, p.586-595, October 21-25, 1995.

Example 2: L. Becker and R.H. G*ü*ting, Rule-based optimization and query processing in an extensible geometric database system. ACM Transactions on Database Systems, 17(2):247303, 1992.

References (a part)

- D.S. Batory, J.R. Barnett, J.F. Garza, K.P. Smith, K. Tsukuda, C. Twichell and T.E. Wise, GENESIS: An Extensible Database Management System, IEEE Transactions on Software Engineering, v.14 n.11, p.1711-1730, November 1988.
- M.J. Carey, D.J. DeWitt, D. Frank, M. Muralikrishna , G. Graefe, J.E. Richardson and E.J. Shekita, The architecture of the EXODUS extensible DBMS, Proceedings on the 1986 international workshop on Object-oriented database systems, p.52-65, September 23-26, 1986, Pacific Grove, California, United States.
- N. Derrett and M.-C. Shan, Rule-based query optimization in IRIS, Proceedings of the 17th conference on ACM Annual Computer Science Conference, p.78-86, February 21-23, 1989, Louisville, Kentucky.
- K.R. Dittrich, Object-oriented database systems (extended abstract): the notions and the issues, Proceedings on the 1986 international workshop on Object-oriented database systems, p.2-4, September 23-26, 1986, Pacific Grove, California, United States.
- J.C. Freytag, A rule-based view of query optimization, Proceedings of the 1987 ACM SIGMOD international conference on Management of data, p.173-180, May 27-29, 1987, San Francisco, California, United States.

Example 3: A. Meyer, Pen computing: a technology overview and a vision. SIGCHI Bulletin, 27(3):4690, 1995.

References (a part)

- S.L. Miertschin and C.L. Willis, Mobile computing in the freshman computer literacy course what impact?, Proceedings of the 5th conference on Information technology education, October 28-30, 2004, Salt Lake City, UT, USA.
- A.C. Long, Jr., Improving gestures and interaction techniques for pen-based user interfaces, CHI 98 conference summary on Human factors in computing systems, p.58-59, April 18-23, 1998, Los Angeles, California, United States.
- S. Chatty and P. Lecoanet, Pen computing for air traffic control, Proceedings of the SIGCHI conference on Human factors in computing systems: common ground, p.87-94, April 13-18, 1996, Vancouver, British Columbia, Canada.
- I. Poupyrev, M. Okabe and S. Maruyama, Haptic feedback for pen computing: directions and strategies, CHI '04 extended abstracts on Human factors in computing systems, April 24-29, 2004, Vienna, Austria.
- R. Plamondon and S.N. Srihari, On-Line and Off-Line Handwriting Recognition: A Comprehensive Survey, IEEE Transactions on Pattern Analysis and Machine Intelligence, v.22 n.1, p.63-84, January 2000.

Appendix D

List of Publications

International Journals

- Kritsada Sriphaew and Thanaruk Theeramunkong, Fast Algorithms for Mining Generalized Frequent Patterns of Generalized Association Rules. IEICE Transactions on Information and Systems, Vol.E87-D No3, March 2004. pp. 761-770 (10 pages).
- Kritsada Sriphaew and Thanaruk Theeramunkong, Quality Evaluation for Document Relation Discovery using Citation Information. IEICE Transactions on Information and Systems (11 pages) (to be appeared).
- Kritsada Sriphaew and Thanaruk Theeramunkong, Universal Frequent Itemset Mining for Discovering Document Relations Among Scientific Research Publications. Submitted to Data & Knowledge Engineering (23 pages).

Lecture Notes

- Kritsada Sriphaew and Thanaruk Theeramunkong, Mining Generalized Closed Frequent Itemsets of Generalized Association Rules. Lecture Notes in Artificial Intelligence; Edited by J.G. Carbonell and J. Siekmann, Knowledge-Based Intelligent Information and Engineering Systems, 2003, pp. 476-484 (9 pages).

International Conferences

- Kritsada Sriphaew and Thanaruk Theeramunkong, Measuring the Validity of Document Relations Discovered from Frequent Itemset Mining. Proceedings of the IEEE Symposium on Computational Intelligence and Data Mining (CIDM 2007), April 2007, Hawaii, USA, pp. 293-299 (7 pages).
- Kritsada Sriphaew and Thanaruk Theeramunkong, Revealing Topic-based Relationship Among Documents using Association Rule Mining. Proceedings of the 23'rd IASTED International Muti-Conference on Applied Informatics: Artificial Intelligence and Applications, February 2005, Innsbruck, Austria, pp. 112-117 (6 pages).
- Kritsada Sriphaew and Thanaruk Theeramunkong, A New Method for Finding Generalized frequent Itemsets in Generalized Association Rule Mining. Proceedings of the seventh International Symposium on Computers and Communications, July 2002, Taormina-Giardini Naxos, Italy, pp. 1040-1045 (6 pages).

- Kritsada Sriphaew and Thanaruk Theeramunkong, A New Set Enumeration for Mining Frequent Itemsets in Generalized Association Rule Mining. Proceedings of the International Symposium on Communications and Information Technologies 2001, November 2001, Chiangmai, Thailand, pp. 25-28 (4 pages).