DEPAUL UNIVERSITY

UNIVERSITY LIBRARIES

DePaul University

Via Sapientiae

College of Computing and Digital Media
Dissertations

College of Computing and Digital Media

Spring 2009

Improving Automated Requirements Trace Retrieval Through Term-Based Enhancement Strategies

Xuchang Zou
DePaul University

Follow this and additional works at: https://via.library.depaul.edu/cdm_etd

Part of the Software Engineering Commons

IMPROVING AUTOMATED REQUIREMENTS TRACE RETRIEVAL

THROUGH TERM-BASED ENHANCEMENT STRATEGIES

BY

XUCHANG ZOU

A DISSERTATION SUBMITTED TO THE SCHOOL OF COMPUTING,

COLLEGE OF COMPUTING AND DIGITAL MEDIA OF

DEPAUL UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE

OF

DOCTOR OF PHILOSOPHY

DEPAUL UNIVERSITY

CHICAGO, ILLINOIS

2009

DePaul University
College of Computing and Digital Media

Dissertation Defense Report

I have read the dissertation written by:
Name _____   SSN _____

(**To the advisor**):   The following dissertation title is identical to the one on the title page of the draft returned to the student.   This title is approved by me and it is to be used when the final copies of the dissertation are prepared.

Title of dissertation:

_____

_____

Advisor's Initials _____

❑   Acceptable.   Candidate may proceed to make final copies

❑   Pass, with revisions stated below:

_____

_____

_____

❑   Not Acceptable.   Please explain:

_____

_____

_____

_____		
Advisor (Print Name)	Signature	Date
_____		
1st Reader (Print Name)	Signature	Date
_____		
2nd Reader (Print Name)	Signature	Date
_____		
3rd Reader (Print Name)	Signature	Date
_____		
4th Reader (Print Name)	Signature	Date

ABSTRACT

Requirements traceability is concerned with managing and documenting the life of requirements. Its primary goal is to support critical software development activities such as evaluating whether a generated software system satisfies the specified set of requirements, checking that all requirements have been implemented by the end of the lifecycle, and analyzing the impact of proposed changes on the system.

Various approaches for improving requirements traceability practices have been proposed in recent years. Automated traceability methods that utilize information retrieval (IR) techniques have been recognized to effectively support the trace generation and retrieval process. IR based approaches not only significantly reduce human effort involved in manual trace generation and maintenance, but also allow the analyst to perform tracing on an "as-needed" basis.

The IR-based automated traceability tools typically retrieve a large number of potentially relevant traceability links between requirements and other software artifacts in order to return to the analyst as many true links as possible. As a result, the precision of the retrieval results is generally low and the analyst often needs to manually filter out a large amount of unwanted links. The low precision among the retrieved links consequently impacts the usefulness of the IR-based tools. The analyst's confidence in the effectiveness of the approach can be negatively affected both by the presence of a large number of incorrectly retrieved traces, and the number of true traces that are missed. In this thesis we present three enhancement strategies that aim to improve precision in trace retrieval results while still striving to retrieve a large number of traceability links. The three strategies are:

1) Query term coverage (TC)

This strategy assumes that a software artifact sharing a larger proportion of distinct words with a requirement is more likely to be relevant to that requirement. This concept is defined as query term coverage (TC). A new approach is introduced to incorporate the TC factor into the basic IR model such that the relevance ranking for query-document pairs that share two or more distinct terms will be increased and the retrieval precision is improved.

2) Phrasing

The standard IR models generate similarity scores for links between a query and a document based on the distribution of single terms in the document collection. Several studies in the general IR area have shown phrases can provide a more accurate description of document content and therefore lead to improvement in retrieval [21, 23, 52]. This thesis therefore presents an approach using phrase detection to enhance the basic IR model and to improve its retrieval accuracy.

3) Utilizing a project glossary

Terms and phrases defined in the project glossary tend to capture the critical meaning of a project and therefore can be regarded as more meaningful for detecting relations between documents compared to other more general terms. A new enhancement technique is then introduced in this thesis that utilizes the information in the project glossary and increases the weights of terms and phrases included in the project glossary. This strategy aims at increasing the relevance ranking of documents containing glossary items and consequently at improving the retrieval precision.

The incorporation of these three enhancement strategies into the basic IR model, both individually and synergistically, is presented. Extensive empirical studies have been conducted to analyze and compare the retrieval performance of the three strategies. In addition to the standard performance metrics used in IR, a new metric *average precision change* [80] is also introduced in this thesis to measure the accuracy of the retrieval techniques. Empirical results on datasets with various characteristics show that the three enhancement methods are generally effective in improving the retrieval results. The improvement is especially significant at the top of the retrieval results which contains the links that will be seen and inspected by the analyst first. Therefore the improvement is especially meaningful as it implies the analyst may be able to evaluate those important links earlier in the process.

As the performance of these enhancement strategies varies from project to project, the thesis identifies a set of metrics as possible predictors for the effectiveness of these enhancement approaches. Two such predictors, namely *average query term coverage* (QTC) and *average phrasal term coverage* (PTC), are introduced for the TC and the phrasing approach respectively.

These predictors can be employed to identify which enhancement algorithm should be used in the tracing tool to improve the retrieval performance for specific documents collections. Results of a small-scale study indicate that the predictor values can provide useful guidelines to select a specific tracing approach when there is no prior knowledge on a given project.

The thesis also presents criteria for evaluating whether an existing project glossary can be used to enhance results in a given project. The project glossary approach will not be effective if the existing glossary is not being consistently followed in the software development. The thesis therefore presents a new procedure to automatically extract critical keywords and phrases from the requirements collection of a given project. The experimental results suggest that these extracted terms and phrases can be used effectively in lieu of missing or ineffective project glossary to help improve precision of the retrieval results.

To summarize, the work presented in this thesis supports the development and application of automated tracing tools. The three strategies share the same goal of improving precision in the retrieval results to address the low precision problem, which is a big concern associated with the IR-based tracing methods. Furthermore, the predictors for individual enhancement strategies presented in this thesis can be utilized to identify which strategy will be effective in the specific tracing tasks. These predictors can be adopted to define intelligent tracing tools that can automatically determine which enhancement strategy should be applied in order to achieve the best retrieval results on the basis of the metrics values. A tracing tool incorporating one or more of these methods is expected to achieve higher precision in the trace retrieval results than the basic IR model. Such improvement will not only reduce the analyst's effort of inspecting the retrieval results, but also increase his or her confidence in the accuracy of the tracing tool.

ACKNOWLEDGMENTS

I would like to express my deep and sincere gratitude to my advisor, Dr. Raffaella Settimi, who has been a great inspiration during my doctoral studies. I am grateful to her for her constant support and guidance through my research, exams and career choices. Without her help I would not be where I am today.

I would also like to thank my co-advisor, Dr. Jane Huang for sharing her knowledge and thoughts on Requirements Engineering research. I have been fortunate to work with her and Dr. Settimi from day one of my doctoral program. Her comments on my research ideas and approaches have been especially valuable to me. The work would not have been possible without the help from both Dr. Settimi and Dr. Huang.

I also thank my committee members, Dr. Merlin Dorfman, Dr. Bamshad Mobasher and Dr. Noriko Tomuro, for their effort and support.

I would like to acknowledge NSF for funding my initial work during the first two years of my research, and College of CDM for offering me the scholarship during the later time.

Special thanks go to CDM staff, especially Mr. John Glatz, for the support.

I would also like to thank my fellow graduate students whom I have worked with in our Applied Requirements Engineering Lab: Chuan Duan, Eli Romanova and Kwan Linpsangsri. My thanks also go to my dear friends Julie Zhang and Jingli Zhang for their friendship and support.

Lastly, I would like to thank my family for their persistent support. I thank my husband Sha for his love and care.

I dedicate this thesis to my parents.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

GLOSSARY

Some concepts and terminology used in this thesis are listed as follows:

- **Distinct terms**: A list of terms that does not contain any repetition. For example, a document consisting of six terms "*aabcdd*" contains four distinct terms "*abcd*".

- **Nominal compounds**: A sequence of two or more nouns related through modifications, such as "*road condition report*" and "*weather station*".

- **Noun phrase**: a phrase composed of a head noun and its modifiers, such as adjectives, verbs and other nouns. Examples of adjective-noun phrase include "*parallel algorithm*". Examples of noun-noun phrase include "*information retrieval*".

- **Phrase**: a sequence of two or more words related through modification, such as "*parallel and sequential algorithm*" and "*the structure, analysis, organization, storage, searching, and retrieval of information*".

- **Query length**: Number of terms contained in a query.

- **Stop words**: Very common words such as "*a*", "*the*", "*be*", "*as*" that are ignored or filtered out of the document collection textual information.

- **Terms**: Non-stop words that are extracted from the document collection and stemmed to the grammatical roots.

- **Term coverage**: The percentage of query terms occurring in the document the query searches against. As an example, the term coverage between a query "*abc*" and a document "*cdefg*" that shares one query term "*c*" is 1/3.

- **tf-idf**: A standard term weighting scheme where *tf* stands for term frequency and *idf* stands for inverse document frequency. A term is assigned a relatively high weight if it occurs many times in the document, and is contained in a small number of documents.

CHAPTER 1: INTRODUCTION

1.1 Background

Requirements traceability is defined as "*the ability to describe and follow the life of a requirement, in both a forwards and backwards direction, i.e., from its origins, through its development and specification, to its subsequent deployment and use, and through periods of ongoing refinement and iteration in any of these phases*" [28]. It is concerned with managing and documenting the life of requirements. The main goals of requirements traceability are to ensure that a generated software system satisfies the specified requirements, that all requirements have been implemented by the end of the lifecycle, and also to analyze the impact of proposed changes on the system. In other words, providing requirements traceability not only helps the verification and validation of requirements, but also supports some other critical software engineering activities such as change management and impact analysis in evolving software systems.

Four types of requirement traces are typically defined with respect to their process relationship to a requirement's life [14]:

- *Forward from requirements*: Requirements must be assigned to system components such that the accountability is established and the change impact of requirements can be evaluated.

- *Backward to requirements*: Compliance of the system with the requirements must be verified.

- *Forward to requirements:* Changes in stakeholder needs, as well as in technical assumptions, may require a radical reassessment of requirements relevance.

- *Backward from requirements:* The contribution structures that model the participants in the requirements generation are crucial in validating requirements.

The first two trace types refer to *post- traceability* that links requirements to the design and implementation, documents responsibility assignment or system compliance verification. The last two trace types refer to *pre-traceability* that documents the context and rationale from which requirements are generated. Although it provides demanded links between

business and IT, pre-traceability is often less understood than post-traceability and the existing support for pre-traceability is often considered to be inadequate [27].

Early work in the traceability area has focused on identifying effective strategies for requirements traceability. As early as 1978, Pierce [53] utilized a requirements database to facilitate requirements analysis to aid the verification and validation of a Navy undersea acoustic sensor system. Hayes [32] developed a front end for a validation and verification system that was based on a relational database. This front end provides functionalities for extracting requirements text and assigning keywords in order to support keyword matching between requirements.

Other aspects of traceability such as tracing process and the requirements change management also have been investigated in earlier work. Gotel and Finkelstein [27] conducted a survey on industrial practitioners and commercial/research tracing tools to analyze the requirements traceability problem. They analyzed the difference between pre-specification and post-specification traceability and suggested that improving pre-specification traceability is critical to improve the requirements traceability process. Pohl [54] proposed an approach to achieve pre-traceability by utilizing a three-dimensional framework for a requirements engineering trace repository to enable both selective trace retrieval and automated trace capture. Ramesh [56] investigated the importance of requirements traceability from the practitioner point of view and identified two groups of users with respect to their traceability practice. The two distinct groups are: low-end users who see traceability simply as a mandate from the project sponsors or for compliance with standards, and high-end users who see traceability as a major opportunity for customer satisfaction and knowledge creation throughout the systems lifecycle. In [58] they further developed requirements traceability reference models for low-end and high-end groups individually and described ways to migrate from one to another.

The aspect of requirements change tracing was also investigated by several researchers. Cleland-Huang et al. [7] proposed a technique named Event-Based Traceability for supporting performance related impact analysis. Their method, which is based upon the event-notifier design pattern,

establishes and utilizes traceability links between requirements and performance models by creating loosely coupled links through the use of publish-subscribe relationships between dependent objects. Ramesh and Dhar [56] developed a conceptual tool named REMAP (REpresentation and MAintenance of Process knowledge) that utilizes process knowledge, which captures the historical information of design decisions made at the early stage of the system lifecycle, to manage changes in the system and maintain communication between different teams.

Table 1.1: An example of a RTM (Requirement Traceability Matrix)

Requirements	UML Classes				
	C1	C2	C3	C4	C5
R1	×				
R2		×		×	
R3	×				
R4			×		×
R5				×	

Research has also been conducted on the identification of traceability rules. Spanoudakis et al. [71, 72, 73] presented a system for automating traceability links generation between textual requirement artifacts and object models such as classes, attributes and operations using heuristic traceability rules. Three types of beliefs were introduced and measured: beliefs in correctness of the traceability rules, beliefs in correct traceability relations generation, and beliefs in the satisfiability of traceability rules. Egyed et al. presented a Trace Analyzer technique for automating requirements tracing [18, 19, 20]. Their approach defines trace dependencies through transitive reasoning and the shared use of a "common ground" such as source code. A graph is then built based on the common ground and its overlap with the artifacts. The trace analysis is an iterative process using large amount of rules to manipulate the graph structure. Trace dependency is established if two artifacts relate to at least one common node in the graph.

Various commercial tools and research prototypes have been developed to assist requirements traceability. These supporting tools employ a variety of techniques such as cross references [22], traceability matrices [15], hypertext [42] and templates [39]. As an example, Table 1.1 displays a traceability

matrix, a table in which the identifiers of software artifacts are placed in the left column and the top row. A mark is placed in the intersecting cell of a row and a column to indicate the relationship between the two corresponding artifacts. DOORS [75] and some other tools include requirements traceability matrices to establish, maintain and report on traceability between any type of software artifacts including for instance stakeholder requests, requirements specifications and test cases. Kaindl [42] proposed to utilize hypertext to represent the relationships among requirement statements and the representation of objects in a domain model. In their tool RETH (Requirements Engineering Through Hypertext) requirements are represented using frames and the links are explicitly provided to the user as hypertext nodes in the frames.

Depending on the specific problem they tackle, these tools provide a useful framework to assist the analyst in establishing and maintaining traces between specific information types. However, all these tools have the biggest shortcoming: they still require intensive effort by the analyst. As an example, tools embodying traceability matrices require the analyst to manually create links between the requirements and the software artifacts of interest, and to manually maintain the matrix in a timely manner to reflect the system changes during its evolution. This clearly imposes a huge burden on the analyst especially when the system grows.

To address the problem of the time consuming process for identifying potential links, in the past few years some researchers have applied Information Retrieval (IR) techniques to dynamically generate traceability links on an "as-needed" basis [1, 2, 8, 9, 33, 34, 35, 46, 47, 69]. In a typical tracing scenario, the analyst examines a requirement and searches through the software artifacts in the system to identity the ones that satisfy this requirement or will be potentially impacted by any change in this requirement. The process bears a significant resemblance to the IR tasks in which the search engine analyzes the user issued query and evaluates which documents in the whole collection are relevant to the information conveyed in that query. Software artifacts such as requirements, design documents and source code contain large amount of textual information. For this reason, IR techniques can be employed effectively to search for traces between requirements and other

software artifacts.

Figure 1.1: Screenshot of Poirot: An automated requirements tracing tool

As an example, Figure 1.1 displays the interface of the IR-based requirements tracing tool called Poirot:Tracemaker [9, 43, 69] that was developed by our research group. In this screenshot a requirement is issued as the query by the analyst and a set of UML classes are the searchable documents. In practice, the query and the searchable documents could be any types of software artifacts in the system. This tool implements a Probabilistic Network (PN) model to evaluate the relevance between the query and the traced documents (The PN model will be described in detail in section 2.3). After the relevance scores have been generated, the traced documents are ranked in decreasing order of their relevance scores. By applying a threshold, a set of candidate links that score over the threshold is returned to the analyst for further evaluation. Traced documents that are below the threshold are considered unlikely links and will not be shown to the analyst directly.

IR-based approaches eliminate the upfront effort of establishing and

maintaining a traditional traceability infrastructure such as a matrix or a set of hyperlinks. Recently published research results have shown the great potential of IR based automated traceability tools to augment traditional tracing methods, i.e. to help them construct and maintain trace matrices [1, 2, 8, 9, 33, 34, 35, 46, 47, 69]. These approaches will be fully described in chapter 2 of the thesis.

1.2 Precision Problem in Automated Trace Retrieval

The effectiveness of a requirements tracing tool is typically assessed using two standard IR metrics: *recall*, the proportion of true links that are retrieved by the tool out of all true links available, and *precision*, the proportion of true links in the set of retrieved links [25]. The formulas for calculating these two metrics are defined in section 3.2.

There is usually a trade-off between recall and precision. An attempt to increase recall often results in retrieving more false positives (incorrect links) which consequently decreases precision, while an attempt to improve precision may lead to retrieving less true positives (correct links) which decreases recall.

Search engines in the general IR field usually prefer precision over recall. For instance, a user using a web search engine, such as Google or Yahoo, would probably only look at the top 20 retrieved web documents to see if they are sufficient for his or her information needs, not caring about whether some relevant documents are missing. However in the context of requirements traceability, a tracing tool must obtain high recall, i.e. it must retrieve as many true links as possible in order to be effectively used by an analyst. Compared to the time-consuming, error-prone manual search throughout the whole software artifact collection to identify missed traces, a requirement analyst would rather filter out unwanted links in the candidate links list as it involves significantly less effort.

As a result of pursuing high recall, low precision in the retrieval results has become a concern for automated tracing tools. Previous empirical studies in dynamic tracing retrieval indicate that when a high recall level of 90% is achieved, precision is usually below 40% and sometimes is even worse than 10% [1, 2, 8, 9, 33, 34, 35, 46, 47, 69], regardless of the IR models adopted in the tracing tools. Although using an automated IR approach significantly

reduces the effort required to manually perform a trace, the low precision of the results means that analysts would still have to evaluate a considerably large number of returned links in order to differentiate between true and false traces. In fact, at a precision level of 10%, an analyst would have to look on average through ten candidate links in order to find a good one. Although this is a significant improvement over a brute-force analysis, it still requires significant effort by the analyst and introduces the possibility of human error. The low precision of the trace retrieval results is likely to negatively affect the analyst's belief in the accuracy of the tool. As a result, industrial practitioners may hesitate to adopt IR-based automated traceability tools.

1.3 Methods and Contribution

The need to address the low precision problem in the automated traceability has motivated the research work presented in this thesis. More specifically, the research described in this thesis investigates several enhancement strategies to improve the trace retrieval methods with the main goal to increase the precision in the retrieval results while maintaining high recall values.

The Probabilistic Network (PN) model [77], an IR model which is fully described in section 2.3, provides a baseline against which the enhancement strategies are compared. In order to be useful to the analyst, the automated requirements traceability tools must retrieve as many true links as possible. In our empirical studies, an objective function was therefore established for all the retrieval algorithms to achieve high recall levels at 90% and 80% on most of the datasets (for datasets on which such high recalls can not be achieved, the objective function was to obtain recall as high as possible). An exploratory study described in Chapter 4 analyzes the performance of the basic probabilistic network model in retrieving traceability links between requirements and various software artifacts. This study identifies several factors that may cause the retrieval of incorrect links, or the omission of correct links. These results are used to develop enhancement algorithms designed to utilize one or more of the identified factors in order to improve the retrieval precision of the basic probabilistic network model. The results are also used to develop predictors of the enhancement strength of the new algorithms by evaluating certain characteristics of a specific software system.

Although *tf-idf* is one of the most commonly used term weighting schemes in IR, numerous studies in this field have suggested that the standard *tf-idf* method can be improved by considering some characteristics of a specific document collection [41, 70]. The research work presented in this thesis is therefore specifically focused on investigating term-based enhancement strategies to optimize the standard *tf-idf* term weighting scheme employed in the context of the requirements traceability problem.

Three such enhancement strategies are explored in this thesis:

1) *Query Term Coverage (TC)*

TC approach utilizes the concept of *query term coverage* which is defined as the extent to which terms in the query co-occur in the searched document. The approach assumes that documents containing a larger percentage of distinct query terms are more likely to be relevant to the query. Therefore, incorporating the TC factor into the basic PN model is expected to increase the probabilities of true links more significantly than false links. The change can enable us to set a higher threshold to filter out more false links in the retrieval results and consequently increase the precision of the retrieved links.

2) *Phrasing*

The use of phrases in the general IR context has been studied extensively. Compared to the use of single words (terms), phrases are usually considered more accurate at describing the content of a document and therefore can help improve the accuracy of the text retrieval. Defined over a space of single terms, our basic PN model often retrieves some irrelevant documents and this contributes to the low precision problem. The thesis introduces a new approach to incorporate phrasing techniques into the PN model with the goal of reducing the number of irrelevant documents that are retrieved and therefore improving the overall precision of the results.

The TC approach and the phrasing pursue similar goals and their application overlaps to a certain degree as the shared terms between a query-document pair could contain both single terms and phrases. The current definition of TC in this thesis does not differentiate between phrases and single terms among the shared terms. As an example, assume that a query "*A road section shall be added*" shares three distinct single terms "*road*", "*section*" and "*add*" with a certain document. The TC approach would treat all

the three terms as single terms in the computation of the TC factor. However, compared to the term "*add*", we believe terms "*road*" and "*section*" which form a phrase "*road section*" provide a stronger indication that the query and the document should be related. We therefore keep the concept of phrasing separated from the query term coverage to stress the importance of phrases in the relevance evaluation between query-document pairs.

3) *Utilizing a project glossary*

A project glossary defines some key terms and phrases which capture the critical meaning of the software project. We propose to strengthen the contribution of these important glossary items to the computation of the probability scores by increasing their weights. The incorporation of this approach into the PN model aims at increasing the probability score of links sharing glossary items as these links are assumed more likely to be true links. The change can potentially improve the precision by filtering out more false links.

Figure 1.2: Applying enhancement strategies to improve basic retrieval results
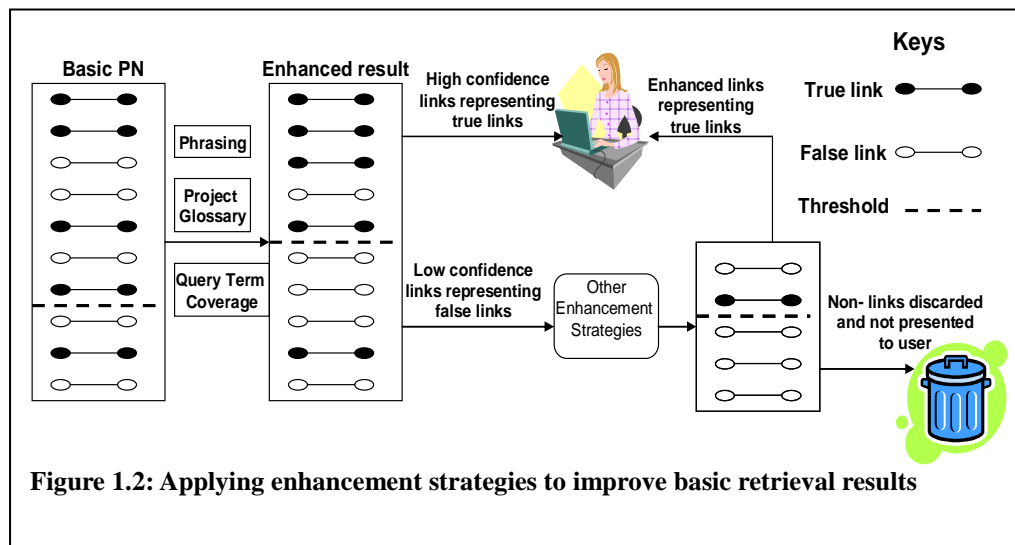
The work introduced in this thesis supports the development and application of automated tracing tools. The three strategies share the same goal of improving precision in the retrieval results to address the low precision problem, which is a big concern associated with the IR-based tracing methods. As depicted in Figure 1.2, a tracing tool incorporating one or more of these methods is expected to return a candidate links list containing a smaller number of false positives than using the basic PN model. Such an

improvement will not only reduce the analyst's effort in inspecting the retrieval results, but also increase his or her confidence in the accuracy of the tracing tools and consequently help to build the analyst's confidence in the tools.

An analysis is conducted to study which project characteristics have a more significant influence on the effectiveness of each of the three enhancement algorithms. Such a study defined a set of project-level factors that can be used to identify which algorithm will be effective in trace retrieval for a given project. Two project-level predictors, namely *average query term coverage* (QTC) and *average phrasal term coverage* (PTC), are presented. These metrics can be adopted to define intelligent tracing tools that can automatically determine which enhancement strategy should be applied to achieve the best retrieval results on the basis of the metrics values. In addition, a set of criteria is proposed to evaluate the usefulness of an existing project glossary in improving the trace retrieval results. As an existing project glossary will not be useful for the purpose of trace retrieval if it has not been consistently used in the development process, a procedure is also presented to automatically extract critical keywords and phrases from the requirements collection of a given project. The extracted items are then used to enhance the automated trace retrieval algorithm. To our best knowledge, the aspect of effectiveness predictors and evaluation for enhancement methods has never been previously investigated in the automated traceability field. The work presented in this thesis will contribute to the traceability study and help improve the adaptability of automated traceability methods.

1.4 Outline of Thesis

The thesis is organized as follows:

Chapter 2 introduces and compares three IR models most frequently adopted in automated traceability: the Vector Space Model (VSM) [65], the Latent Semantic Indexing (LSI) model [81], and the Probabilistic Network (PN) model [77]. To address the low-precision problem associated with the IR-based automated tracing tools the distribution of true links among the retrieved traces is analyzed empirically and reveals interesting patterns that will be used in the next sections to define effective enhancement methods to improve the retrieval accuracy. This analysis also leads to the development of

a confidence-score mechanism to help the analyst in evaluating the retrieval results. Several enhancement methods proposed previously to improve trace retrieval accuracy are also reviewed in Chapter 2.

Chapter 3 describes five datasets that will be extensively used in the empirical studies in the following chapters. The metrics for measuring the accuracy of the retrieval results, including standard IR metrics of recall and precision, as well as some other metrics used in our studies are also defined in this chapter.

Chapter 4 through 6 fully describe the three enhancement strategies: *TC*, *Phrasing* and *Project Glossary*. Each strategy is first defined, and the incorporation of the strategy into the basic PN model either individually or synergistically is introduced, followed by the description and discussion of the evaluation of each new retrieval algorithm in various datasets.

Chapter 7 identifies a set of metrics and the project characteristics that can be used as predictors of the effectiveness of the enhancement approaches. The prediction model utilizing such predictors is then introduced and evaluated in a small-scaled empirical study. In addition, an iterative technique is also described to determine which trace retrieval approach is more effective on a new project when no prior knowledge on traces is available. This technique builds a partial answer set containing information collected from user's feedback.

In Chapter 8 a set of criteria for useful project glossaries for the trace retrieval purpose is presented. A method to automatically extract keywords and phrases from the requirements collection is also introduced and evaluated. This automated method is designed to be used in lieu of a missing or ineffective project glossary.

The final chapter summarizes and discusses the research work presented in this thesis.

CHAPTER 2: IR-BASED AUTOMATED REQUIREMENTS TRACING

Automated requirements trace retrieval techniques that utilize IR methods are typically comprised of the following steps. First of all the textual information contained in the software artifacts collection is pre-processed by removing stop words such as articles, pronouns, and conjunctions that are very common and therefore not useful for retrieval purposes. Each remaining word is then stemmed to its grammatical root by eliminating suffixes and prefixes. Based on the resulting set of terms, relevance scores between requirements and traceable software documents are evaluated using a specific IR model. Requirement-document pairs that score over a certain threshold are then considered candidate links and returned for evaluation to the analyst in decreasing order of their relevance scores.

Three commonly used trace retrieval models that have been applied most frequently in traceability research studies are the Vector Space Model (VSM), Latent Semantic Indexing (LSI) model, and Probabilistic Network (PN) model. Several requirements tracing prototype tools employing different IR models have been developed to assist the analyst in the tracing tasks [1, 2, 8, 9, 33, 34, 35, 44, 46, 47, 69]. Based on the published research results, none of these IR models have shown consistently better performance than others in requirements trace retrieval.

These three IR models will be fully described and discussed in the first three sections of this chapter. As part of the study on the low-precision problem associated with the IR-based automated tracing tools, section 2.4 describes an analysis of the links distribution in the retrieval results using the PN model and presents a confidence-score mechanism to help the analyst in evaluating the retrieval results. The last section introduces and analyzes five enhancement methods previously proposed to improve the low precision in the automated trace retrieval results.

2.1 Vector Space Model (VSM)

The VSM [65, 66, 68] represents each query and each searched document as term vectors defined in the space of all terms $T = \{t_1, t_2, ..., t_n\}$ that are extracted from the whole document collection. Formally a document d can be represented as a vector $\vec{d} = (w_{1,d}, w_{2,d}, ..., w_{n,d})$, where $w_{i,d}$ is the term weight

associated with the term i with respect to d, and n is the total number of terms in the term space. Similarly a query q can also be represented as a vector $\vec{q} = (w_{1,q}, w_{2,q}, ..., w_{n,q})$.

Each term t is assigned a weight that reflects its importance to represent the contents of the query or the document. Such term weights are usually computed using a standard term weighting scheme known as *tf-idf* [Salton & Buckley, 1988] where *tf* stands for term frequency and *idf* stands for inverse document frequency. The *tf* of a term t_i with respect to a document d is usually computed as $tf(t_i, d) = \dfrac{freq(t_i, d)}{|d|}$, where *freq(t$_i$,d)* is the frequency of the term in the document and is normalized by the length of the document. The *idf* of a term t_i is usually computed as $idf_{t_i} = \log_2(\dfrac{n}{n_i})$, where n is the total number of the documents in the whole collection and n_i is the number of documents in which t_i occurs. $w_{i,d}$, the term weight associated with the term i with respect to a document d, is then calculated as $w_{id} = tf(t_i, d) \times idf_{ti}$. A term is considered relevant for representing a document's content and is assigned a relatively high weight if it occurs many times in the document, and is contained in a small number of documents.

The relevance between the query q and the document d is measured by a similarity score *sim(d,q)* that is defined as the cosine of the angle formed by their corresponding vectors, and is computed as:

$$sim(d, q) = \frac{\sum w_{i,d} \times w_{i,q}}{\sqrt{\sum w_{i,d}^2 \times \sum w_{i,q}^2}}$$

VSM has been adopted in automated tracing tools to trace between various types of software artifacts. For example, Hayes et al. [33, 34, 35] built a tool named RETRO (REquirements TRacing On-target) that implements VSM and LSI (LSI will be described in detail in the following section). Its effectiveness in trace retrieval has been evaluated on several datasets focusing on the tracing from high-level to low-level requirements. Our research group has developed a prototype tracing tool Poirot [9, 43, 69] that implements both VSM and PN Model (PN model will be introduced in section 2.1.3) to dynamically generate and retrieve links between requirements and software artifacts including UML

classes, Java code and design documents.

Although VSM is easy to implement and the similarity scores between two vectors are easy to calculate, results from all these studies indicate that in general VSM returns relatively low precision, ranging from 7-14% when a high recall value of 90% is achieved. The unsatisfying retrieval performance of VSM is often caused by strict word-matching. Since there are usually many ways to express a given concept (synonyms), terms in a relevant document might not match the query. Additionally, as words could have multiple meanings (polysemy), an irrelevant document may be incorrectly retrieved if it uses the same terms with the query.

2.2 Latent Semantic Indexing (LSI)

As a variation of VSM, LSI is based on concept matching with the purpose to improve simple word-matching. With LSI, relevant documents that have no terms in common with the query, and therefore would be missed by the standard *tf-idf* approach, may still be retrieved. The assumption of this model is that there are some underlying semantic structures (latent structures) in documents and terms which are often hidden behind the various word choices.

LSI applies Singular Value Decomposition (SVD) technique [24] to map terms and consequently query and documents vectors into a lower dimensional space associated with concepts. According to the SVD model, any $a \times b$ matrix X, where a, b are the numbers of rows and columns in X respectively, can be decomposed into the product of three other matrices:

$$X = T_m S_m D_m^T$$

where T_m (a $a \times m$ matrix) and D_m (a $b \times m$ matrix) have orthogonal and unit-length columns, and S_m is a $m \times m$ ($m=min(a,b)$) diagonal matrix of singular values. If we only keep the k (k<m) largest singular values of S_m, and only retain the corresponding k columns in matrix T_m and the k rows in D_m^T, a new matrix X' can be obtained as:

$$X^{'} = T_k S_k D_k^T \approx X$$

X' is the approximated matrix of rank *k* with the best least square fit to the original matrix *X*. That is, by keeping the *k* largest singular values, we are able to capture the major structure of the original data and discard much of the noise that causes poor retrieval performance.

Based on SVD techniques above, LSI has been adopted in IR by firstly constructing a $t \times d$ term-document matrix X from a collection of documents, where t is the total number of the terms in the space and d is the total number of documents. Each element X_{ij} ($0 \leq i < t$, $0 \leq j < d$) in X represents the frequency of term i in document j. After a k-dimensional SVD decomposition, the original term-document matrix is projected into a lower dimensional new space. The relevance between the query and the document vectors is also evaluated by its cosine similarity measure in the space of reduced dimension.

Lucia et al [44, 45] built a tool ADAMS (Advanced Artifact Management System) that implements the LSI model to help software engineers recover traceability links missed in manual tracing. Macus and Maletic [46] utilized LSI to recover links between source code and documentation, including internal documentation such as comments. Although considered to be a more accurate IR model than VSM, LSI has not consistently outperformed VSM in traceability applications. For instance, in the tracing experiments conducted by Hayes et al [35] in which both VSM and LSI were utilized to retrieve links in two datasets, the two IR models have similar performance in one dataset but VSM achieved higher recall and precision in another dataset. However, results by Macus and Maletic [46] showed that LSI was able to achieve higher recall and precision than VSM in the same datasets used in their study. Typically LSI is considered to perform better on larger and textually richer datasets, as the reduction of dimensionality may result in significant information loss for datasets with fewer terms and concepts [81].

An open question of LSI is how to choose the optimal dimension k, such that k is large enough to capture the majority of the latent structure of the document collection, and at the same time small enough to reduce unimportant details. In practice, the dimensionality is often determined experimentally, or selected based on previous work. Another concern of LSI is that the model tends to be computationally intensive for SVD decomposition when the corpus of the document collection grows, as the sparse term-document matrix can become enormous.

2.3 PN (A Probabilistic Network Model)

Our previous work has applied a Probabilistic Network (PN) model to dynamically generate and retrieve links between requirements and software

artifacts. The probabilistic IR approach uses a flexible mathematical framework to model queries and documents as variables in a concept space defined by the keywords in the documents collection. A probabilistic graphical model [51] is used to represent the relationships between queries and documents, where documents, queries and keywords are represented as nodes in a graph and the relevance relationships are represented as arrows linking a query to related keywords and documents. Although the PN model uses a standard *tf-idf* approach to compute the probability of relevance of a document with respect to a query, it can be extended to allow the use of additional information about the documents collection to improve the learning of relevant links.

Following standard IR terminology, the PN model [77] assumes the existence of a set of documents *{q₁, q₂, …, qₘ}* that are regarded as queries and a document collection *{d₁, d₂, …, dₙ}*. The model interprets each query *q* and each document *d* as random variables defined within a concept space defined by the terms *{t₁, t₂, ……, tₖ}* contained in the document collection, and estimates the relevance between *q* and *d* as the probability of a link existing between them. In this thesis the requirements are regarded as queries, and the document collection contains other software artifacts that may be traced to the requirements. Note the model can be used to trace between any types of document sets. The assumption in our probabilistic model is that a software artifact *d* containing terms that co-occur in the given requirement *q* is expected to be a potentially relevant trace for that requirement. Thus a conditional probability value *p(d|q)* for each *q* and *d* is defined as a function of the frequency of terms co-occurring in both *q* and *d,* and is calculated as follows:

$$p(d \mid q) = \frac{p(d,q)}{p(q)} = \frac{\sum_{i}^{t} p(d \mid t_i) \times p(q,t_i)}{p(q)} \tag{2.1}$$

The three components in the formula are defined as $p(d \mid t_i) = \dfrac{freq(d,t_i)}{\sum_{i}^{t} freq(d,t_i)}$,

$p(q,t_i) = \dfrac{freq(q,t_i)}{n_i}$, and $p(q) = \sum_{i}^{t} p(q,t_i)$, where *freq(d, tᵢ)* represents the

frequency of term *tᵢ* in a document *d* and *nᵢ* is the number of documents in the

searchable document collection $\{d_1, d_2, …, d_n\}$ that contain term t_i. Notice that $p(d|q)$ is equal to zero if d and q do not have any terms in common.

Similarly to VSM, the definition of the probabilistic similarity score is also based on the *tf-idf* standard weighting strategy. The first component $p(d|t_i)$ represents the relative frequency of term t_i in document d and increases with the number of occurrence of t_i in d. *idf* is used to reduce the contribution of common terms to the overall probability value between the query and the document.

Given a query q_i, the conditional probability $p(d_j|q_i)$ is computed for each document d_j in $\{d_1, d_2, …, d_n\}$ using formula 1.1. Documents are ranked in decreasing order of the probability scores to the query q_i according to the value of $p(d_j|q_i)$. A high value of $p(d_j|q_i)$ provides strong evidence that a potential link between q_i and d_j exists. A threshold value is typically established, and all documents whose probability values $p(d_j|q_i)$ fall above the threshold are retrieved as potential links to the query q_i.

2.4 Analysis of Links Distribution

This section describes an analysis of the low precision problem for automated tracing tools. The study reveals that the precision of the results vary greatly with the range of the probability scores assigned by the PN based tracing tool. The proportion of true links increase as the probability scores get larger. Based on the analysis of the precision levels for different probability intervals, and to improve the usability of the probabilistic network tool, a confidence score mechanism is designed to support the analyst in the evaluation of the trace retrieval results with low precision. This mechanism has been deployed in our Poirot:Tracemaker tool [43].

A confidence score represents the belief a user has that a retrieved link is in fact a true link. The importance of implementing this concept to support requirements trace retrieval is critical in light of the precision problem affecting trace retrieval.

The link distribution for true-links and false-links is initially learned from three different datasets, including the Ice-Breaker System (IBS), Event Based Traceability (EBT), and Light Control System (LC) which are fully described in section 3.1. In these datasets, similar patterns are observed among the distributions of the probability scores for true links. The results consistently

show that candidate links with larger probability scores are more likely to be true links. On the other hand, the links with very small probability scores are very likely to be non-links. This is clearly depicted in Figure 2.1, which shows the probability score distribution for true links against the one for non-links for the IBS system. In the region in the center, true-links and non-links are mostly cross-distributed, making it difficult to discriminate between these types of links based only on the probability scores.
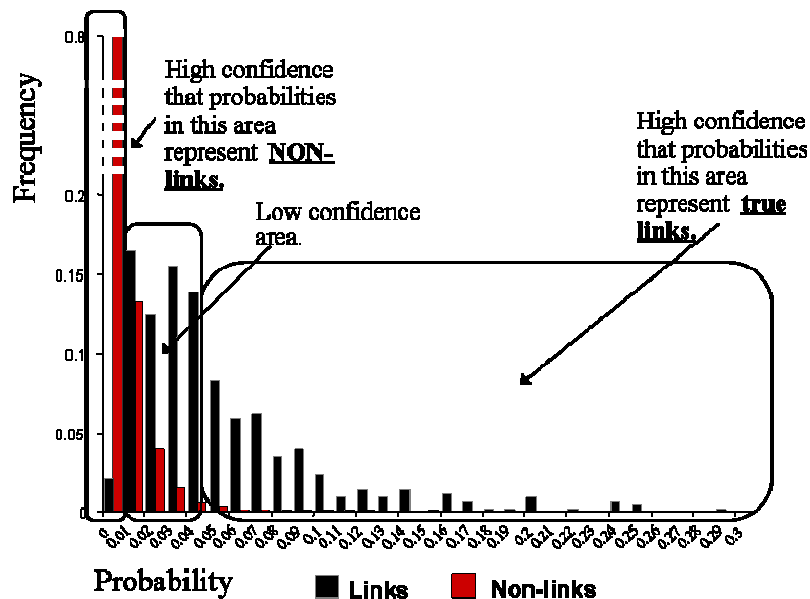
Figure 2.1 Areas of high and low confidence in IBS

Figure 2.2 illustrates the frequencies of true links for the IBS dataset in equally spaced intervals for the probability scores defined in terms of percentile ranks. The graph shows that the frequency of true links increases with the probability scores in a non-linear fashion. In the middle region, the frequency of true links increases at a slower rate than in the top region where the probability values are higher. Similarly for links with very low probability values – i.e. within the region where the true links are scarcer - the frequency of true links increases even more slowly.
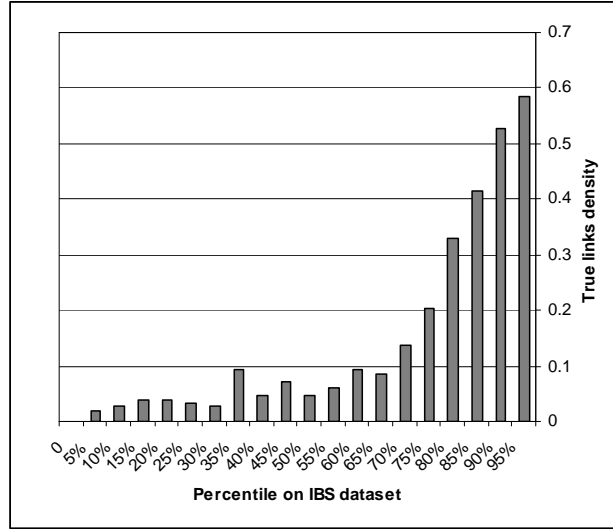
Figure 2.2: Confidence value vs. Probability in IBS

Based on these observations, we defined a segmentation method for computing confidence scores by considering the frequency distribution of true links in different probability intervals. Confidence scores are defined as a measure of belief in the correctness of the results returned by the automated trace retrieval algorithm. For convenience the scores are set between 0 and 100%, and are calculated according to the probability value of a link in respect to the general distribution of probability values for true-links and false-links in the dataset.

The segmentation method divides the range of probability scores in (K+1) intervals and defines a piecewise function that computes confidence scores in each interval according to the frequency of true links in that interval.

As shown in Figure 2.2, true links frequency is relatively high when probability scores are above the 80th percentile, while for scores below the 20th percentile, the frequency of true links is quite low. The distributions of true links for EBT and LC datasets reflect a similar pattern.

The frequency of true links in the *i-th* segment *(p_{i-1}, p_i)* is defined as:

$$\mu_i = \frac{\text{number of true links in } (p_{i-1}, p_i)}{\text{number of candidate links in} (p_{i-1}, p_i)} \qquad (2.2)$$

The measure μ_i can be regarded as the local precision metric relative to the *i-th* segment.

Let C(p) be the confidence score associated to a candidate link with

probability score p. The 100% confidence score is assigned to the candidate link that has probability value equal to or higher than 0.5, i.e. $C(p)=1$ for $p>=0.5$. This is based on our observation that any link with a probability value as high as 0.5 is extremely likely to be a true link. Also we assign zero confidence to pairs (d,q) with zero probability score, i.e. $C(p)=0$ for $p=0$.

Let p_{max} be the maximum probability score returned by the tool. The set of probability scores between $p_0=0$ and $p_{K+1}=p_{max}$ is divided into K+1 intervals. Each interval (p_{i-1}, p_i), for $i=1,...,K+1$, is defined by selecting K percentile values $(p_1, p_2, ..., p_K)$ from the distribution of the probability scores. The K percentile values can be determined according to some prior knowledge of the true links distribution. Such prior knowledge could be based for instance on a set of known traces that can be generated without the need for user evaluation.

If no prior knowledge is available on a certain system, a simpler version of the segmentation algorithm can be proposed using a coarser segmentation and the true links density learned in our previous experiments. However, as users apply the retrieval tool to that system, information on the correctness of some links is obtained through user's feedback, and can be used to learn the confidence values more accurately.

Once the K percentile values have been selected, the average probability $\bar{p}_i$ of the probability scores in (p_{i-1}, p_i) is computed for each interval (p_{i-1}, p_i) for $i=1,...,,K+1$. The confidence score $C(\bar{p}_i)$ is assumed to be proportional to the true link frequency in the i-th interval, and is computed as $C(\bar{p}_i)= \lambda * \mu_i$, where λ is a rescaling factor that can be set by the user and μ_i is defined in expression (2.2).

Each segment in the piecewise function is defined as a straight line joining pairs of points defined as $(\bar{p}_i, C(\bar{p}_i))$ and $(\bar{p}_{i+1}, C(\bar{p}_{i+1}))$ for $i=0,...,K$. The graph in Figure 2.3 shows an example of the piecewise function for a simple case with K=2. In each segment, the confidence value $C(p)$ is computed as a linear function of p with slope depending on the increase in true link frequency between two consecutive intervals.
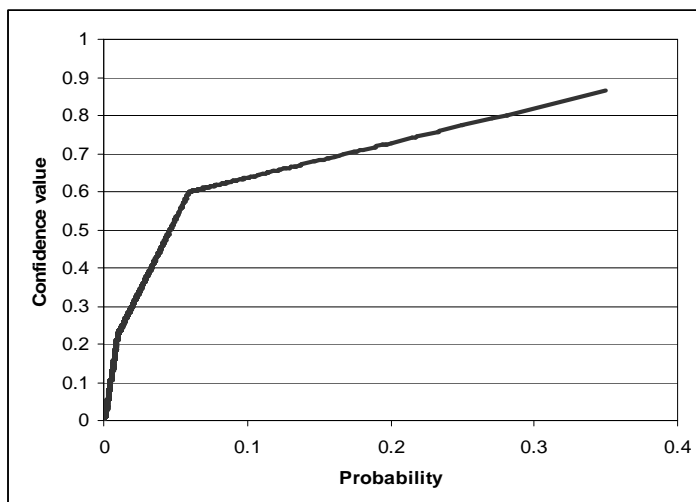
Figure 2.3: Confidence value vs. Probability in IBS

The general expression of the piecewise function for computing confidence values is displayed in Table 2.1.

Table 2.1: Linear stepwise function to compute confidence values

$$C(p) = \begin{cases} \dfrac{C(\overline{p}_1)}{\overline{p}_1} \times p & \text{for } p \in (0, \overline{p}_1) \\[2ex] C(\overline{p}_{i-1}) + \dfrac{C(\overline{p}_i) - C(\overline{p}_{i-1})}{\overline{p}_i - \overline{p}_{i-1}} \times (p - \overline{p}_{i-1}) & \text{for } p \in (\overline{p}_{i-1}, \overline{p}_i), \quad 2 \leq i \leq K+1 \\[2ex] 1 - \dfrac{1 - C(\overline{p}_K)}{0.5 - \overline{p}_K}(0.5 - p) & \text{for } p \in (\overline{p}_{K+1}, 0.5) \end{cases}$$

A small-scale usability study on the effectiveness of this confidence score mechanism was conducted using our Poirot:Tracemaker tool [Lin et al., 2006] to compare the analyst's performance in conducting trace tasks when supported by confidence scores versus tasks that were unsupported. Through observing participants' behavior and as a result of the interview process, we discovered that the confidence scores were very effective in helping the analyst filter out links that were not likely to be true. Analysts preferred to focus their effort in the evaluation of links with higher confidence score, rather than links with very low score. Links with very low confidence score were immediately regarded as not correct. The usability study was fully described in [79].

2.5 Previously Proposed Enhancement Approaches

Prior research in requirements traceability has attempted to improve precision in several different ways. Some approaches adopt standard IR

techniques while others take into account the specific characteristics of the software artifacts collection in the development of their retrieval algorithms. The five previously proposed approaches, namely *thesaurus*, *hierarchical structure information*, *clustering*, *graphical pruning* and *user relevance feedback* are described and discussed in this section.

2.5.1 Thesaurus

Thesauri provide the information about relationships between different terms. The classic IR models only compare the co-occurrence of single terms in the query and the document, regardless of the presence of synonyms and acronyms in the document which implicitly could make the document relevant to the query. However in many situations, a relevant document is not retrieved due to the "*ignorance*" of the IR model that is not able to recognize synonyms. As an example, a document containing "*car speed*" will not be considered relevant to a query describing "*vehicle velocity*" based on the basic IR models, even though it might be a true match to the query.

To address this problem, the thesaurus has become a standard component in tracing tools [33, 34, 35, 43] to capture synonyms or acronyms both in the query and the traced artifact which otherwise might have contained no matching terms. In their tracing tool RETRO, Hayes et al. [33, 34, 35] implemented a thesaurus-based retrieval to extend the underlying basic VSM model by considering the synonyms and matching terms used in technical lingo in which the high level and low level requirements are written. Their thesaurus was constructed by manually extracting these matching terms from the data dictionary and from acronym lists available in the appendices of the requirements documents. In their approach, an analyst manually assigned a similarity coefficient between 0 and 1 to each pair of the matching terms to measure the match relevance based on the analyst's perception. A default value is used if the analyst chooses not to assign such as value. The standard cosine similarity computation between a query and a document receives an additive score from matches defined in the thesaurus. Their evaluation of this thesaurus approach focusing on traces from 19 high level requirements to 50 low level requirements showed that the approach performed quite well. The precision was significantly increased from 11% using the basic VSM model to 41%. The approach has proven to be especially effective in improving the retrieval

results when the queries and the artifacts are written using very different terminology.

However the approach proposed by Hayes et al. [2003; 2006] assumes that there are data dictionaries and the acronym lists available in the appendix of requirement documents from which the thesaurus can be manually extracted by copy and paste. In reality not every project comes with such information. When an appendix is not available, constructing a thesaurus can be time consuming, as it may require extensive knowledge on a specific project in order to identify the matching technical terms.

2.5.2 Hierarchical structure information

Many software artifacts are constructed in a hierarchical structure. Examples include headings for requirements and sub-requirements in which information contained in a head requirement describes the general meaning of its sub-requirements; or packages and classes, where the name of a package indicates the general functionalities of the classes contained in it.

Cleland-Huang et al. [9] proposed utilizing hierarchical information describing interrelationships between ancestors of the query and the traced artifact. They observed that the information contained in the ancestors may provide necessary context to understand its descendants. Their proposed approach therefore enhances the probability value between a query and an artifact if their ancestors share the same terms. The hierarchical approach was evaluated against three datasets [9] tracing from requirements to UML classes. The experiment applied the hierarchical approach either to the entire dataset or to a subset of mid-probability links which based on the links distribution study in section 2.4 are considered low confidence links. However, only one dataset demonstrated an improvement on precision with an increase of 11% at recall of 90% when the approach was applied to the low confidence links. The other two datasets returned minimal or no improvement.

Their analysis revealed that the two datasets in which the hierarchical approach was not effective do not have strong hierarchical information, and the package names were not as meaningful as they could have been. Therefore, strong hierarchical structure is essential for this approach to be effective. Loose hierarchy within the artifacts, which is often presented in software projects, impacts the effectiveness of this enhancement strategy.

2.5.3 Clustering

Clustering is another enhancement algorithm proposed by Cleland-Huang et al. [9] which is based on the observation that true links tend to occur in physical clusters within a hierarchy. These clusters can be logical groups of software products such as a package containing multiple classes. The algorithm hypothesized that if a link is established between a query and a document, and if the document belongs to a logical cluster, then the likelihood of other documents within that cluster also being relevant to the query should be increased. The concept is called *document side clustering*. Similarly, *query side clustering* can also be applied to individual documents and clusters of queries if a link is established between the document and one of the queries within the same cluster.

Experiments were conducted by the authors against the same three datasets used for evaluating the hierarchical approach. For the dataset with relatively strong hierarchical information, a significant improvement was observed only when document-side clustering was applied to the retrieved links that have lower probability scores. Query-side clustering did not achieve any improvement. The two datasets in which the hierarchical structure was not very strong did not return any significant improvement from applying either document-side or query-side clustering. In the experiments document side clusters were formed from groups of lower-level requirements and the document side clusters were classes within packages.

Similarly to the hierarchical approach, the effectiveness of the clustering algorithm in increasing the precision of the retrieval results is strongly affected by the structure of the documents collection such as the decomposition of requirements into sub-requirements, the grouping of classes into packages. Datasets with loose logical structures are most likely unaffected by this algorithm.

2.5.4 Pruning

In the same paper [9], Cleland-Huang et al. proposed a semi-automated graph pruning technique to apply to specific areas in which the precision of the retrieval is particularly low. The approach asks the analyst to provide initial feedback against a training set to identify the groups of requirements and documents that are mostly likely to spawn false links. Constraints are then

established between the groups of requirements and documents such that the probability of the links in the constraining groups is either cut by half or reduced to zero.

As the pruning technique requires a training set to initialize the process, the effectiveness of this approach is greatly affected by the size of the training set. In the experiments conducted by Cleland-Huang et al. [9], promising improvement on precision of the retrieval results was observed only when the training set size was relatively large (50% of the query-document pairs were included in the training set for one of their datasets). It indicates the approach will be effective only if sufficient feedback is obtained from the analyst.

2.5.5 Relevance feedback mechanism

Relevance feedback is a classic IR technique [62] where the search system revises the list of the search results according to the user's agreement or disagreement on the system's evaluation of the document relevance. In a typical relevance feedback process, the user is presented with a list of ranked documents and asked to indicate which documents are relevant to his or her query and which ones are not. The decisions are collected for query extension by typically increasing the weighting of terms that are contained in relevant documents, and decreasing the weight of terms that are in irrelevant documents. The updated query is then used by the system to induce a new set of relevant documents, possibly including new documents. The display, evaluation, and re-ranking cycle repeats until the user is satisfied with the returned results.

Hayes et al. [33, 34, 35] incorporated in their tracing tool RETRO the standard Rocchio algorithm [62] to extend the VSM model. Given a query, the analyst was provided with a "link"/"no link" choice for each candidate link retrieved by the tool. In the experiments they simulated the user feedback by using the pre-defined "answer set" of the datasets. Evaluation was provided on the top i (i=1, 2, 3 or 4) ranked documents in each iteration and the cycle was repeated eight times. The results indicated that utilizing user relevance feedback can potentially increase recall by including the omitted true links, and also improve precision by excluding false positives.

The biggest drawback associated with their proposed feedback mechanism however, is the tedious process of repetitively providing relevance information

on retrieved documents. The results suggested that the approach is not effective when the number of evaluated documents in each cycle is too small, which means the analyst may have to provide considerable amount of input in order to provide sufficient feedback information. In addition, this mechanism tends to only improve the results on the queries upon which the feedback was supplied. The tradeoff between the amount of the input solicited from the user and the improvement on the retrieval results of this approach has not been fully studied.

As described above, although individually proven to be effective in improving the retrieval results to some extent, each of the methods has its own drawbacks. Some methods have strong constraints and may be only applicable to datasets with some specific characteristics. Therefore, the objective of our research work is to investigate effective enhancement strategies that are simple, easy to implement and require the least amount of additional effort from the analyst. Furthermore, the factors that affect the effectiveness of the proposed strategies will also be investigated with the goal to provide metrics for predicting how well a certain strategy may perform in a specific dataset. These metrics can be used to develop tracing tools that can automatically determine whether or not to apply a certain enhancement strategy in order to achieve the best retrieval results according to the metrics values.

CHAPTER 3: DATASETS, METRICS AND BASELINE RESULT

This chapter introduces the five datasets that are used in the empirical studies presented in this thesis to evaluate the trace retrieval approaches. A set of metrics for the evaluation of the retrieval accuracy and effectiveness is also described. Section 3.3 describes the trace results in these five datasets using the basic PN retrieval algorithm which provides a baseline against which the enhancement strategies are compared.

3.1 Test Datasets

Constructing a dataset from a software system for evaluation purpose typically involves the following steps: 1) Extracting textual information of software artifacts which may be specified using various tools, such as MS Word, UML diagrams, Requisite Pro [60] etc. and 2) Manually building a trace matrix which is the answer set explicitly defining the true links between the software artifacts in a system. A trace matrix is the prerequisite for evaluating the retrieval accuracy. Even if the system is supplied with a matrix, additional effort is still needed to validate the accuracy of the given matrix.

Because of the limited availability of such software systems and the effort involved in creating and evaluating the trace matrices, currently we have only five datasets for the empirical studies. These datasets contain a variety of software systems ranging from available research projects to industrial applications or student assignments. The software artifact types contained in these datasets include requirements, design documents, and source code. Therefore, these datasets can be considered representative of the tracing tasks in real software systems.

The details of individual datasets are described below and summarized in Table 3.1 and Figure 3.1:

- **IBS**

IBS (Ice-breaker System) describes the requirements and design of a public-works department system for managing roads de-icing. The system activities include prediction of icy conditions, work order management, truck management, and inventory control. IBS was initially described in "Mastering

the Requirements Process" [99] and then enhanced from materials found at several public works websites. The tracing task in IBS is focused on links between 164 requirements and 71 UML classes. An initial trace matrix was supplied with the system and was also validated by our research group.

IBS is extensively used in our experiments as it is a relatively large dataset with a complete and thoroughly validated trace matrix. In addition, it contains strong hierarchical information in the artifacts, and has a project glossary that was used consistently throughout the writing of the documents and can be used for the evaluation of the glossary strategy discussed in this research proposal.

- **EBT**

The Event-Based Traceability (EBT) system, which was initially developed at the International Center for Software Engineering at the University of Illinois at Chicago [7], provides a dynamic traceability infrastructure based on the publish-subscribe scheme for maintaining artifacts during long-term change maintenance. It is composed of 41 requirements and 52 classes. Its trace matrix was manually built by the developers of the system.

- **LC**

 LC was reconstructed from the well documented Light Control system (Light Control) developed for the University of Kaiserslautern [3]. This system controls the lights in a building based upon user defined lighting schemes, building occupation, and current exterior illumination. Our version of this system consists of 34 requirements and 25 classes. The trace matrix was supplied with the original project.

Each of the matrices for IBS, EBT and LC has been used in our prior work and has therefore undergone a rigorous and lengthy evaluation process.

- **SE450 Student Projects**

 This dataset contains 15 anonymous student term projects for a MS level Software Engineering class at DePaul University. The students were instructed to implement a traffic simulation system using the Java programming language. The implemented system describes vehicles' behavior in a set of roads equipped with traffic lights. The dataset consists of requirements that specify road maps, road segments, vehicles and stoplights, and 15 sets of Java

Table 3.1: Datasets Summary

Dataset	# of requirements	Document type	# of documents	# of true links	Available project glossary
IBS	164	UML classes	71	420	Yes
EBT	41	UML classes	52	135	No
LC	34	UML class	25	91	No
SE450	46	Java classes	475*	1252*	Yes
CM1	235	Low-level requirements	220	361	No

*SE450 dataset aggregated statistics from 15 student projects

source code which vary greatly in structure, as shown in Figure 3.1. The tracing task for this dataset is therefore from the same set of requirements to the 15 sets of Java source code.

As part of the deliverables, each student created a trace matrix for his or her own project to specify one or more of the java classes that implement the individual requirements. All the matrices were independently validated and corrected by two research assistants in our group.

- **CM1**

CM1 is a large dataset extracted from a NASA project for a science instrument and is made available to the public through the Promise Data Repository [55]. The dataset consists of 235 high-level requirements and 220 low-level requirements. A traceability matrix containing 361 true links

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
■ # of class	15	32	33	35	35	26	49	29	26	38	26	29	33	42	27
▨ # of package	5	7	7	5	4	8	6	7	7	5	3	4	4	5	8
▢ # of true links	97	69	103	61	75	70	83	104	84	82	100	87	87	86	64
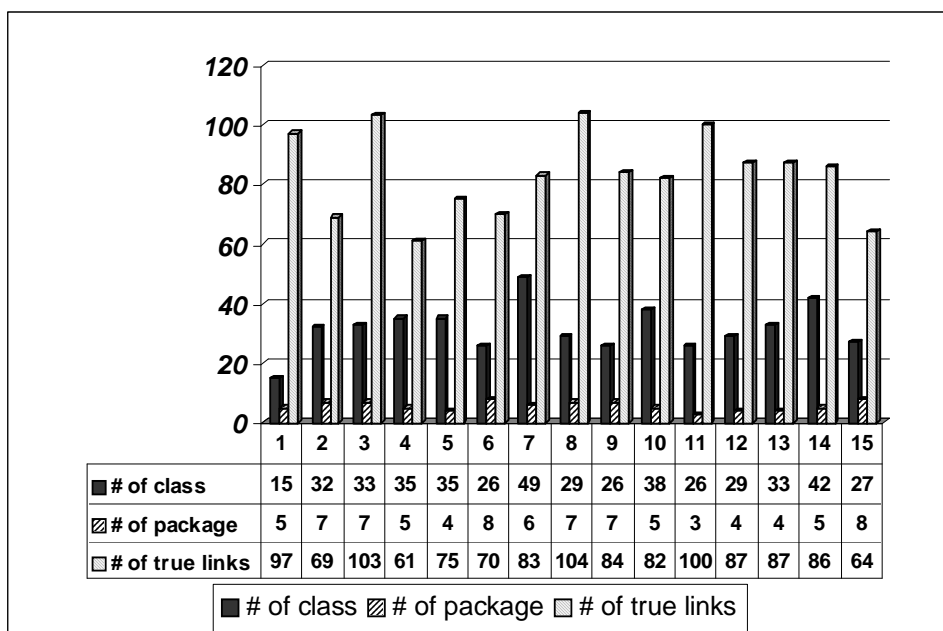
Figure 3.1: Comparison of 15 student projects on SE450 dataset

between high-level and low-level requirements was manually built by NASA and verified by Hayes et al. [33, 34, 35] for their studies on requirements traceability. The tracing retrieval results on CM1 using both VSM and LSI models in their studies have been published in [33, 34, 35].

3.2 Metrics for Retrieval Effectiveness

- **Recall and Precision**

Two standard IR metrics, recall and precision [25], are used as the primary measurements in the evaluation of the accuracy of the retrieval results. Recall measures the proportion of true links that are retrieved out of all true links available, and precision measures the proportion of the true links in the set of retrieved links. They are computed as follows:

$$recall = \frac{\#\ of\ correctly\ retrieved\ traces}{\#\ of\ correct\ traces}$$

$$precision = \frac{\#\ correctly\ retrieved\ traces}{\#\ retrieved\ traces}$$

In tracing retrieval results, precision is usually evaluated at a low threshold in order to achieve high recalls such as 90% or 80%. The precision at these high recall levels represents the overall accuracy in the candidate links list (a set of links that scores above the threshold) returned to the analyst. However in some situations the overall precision alone is insufficient to measure the improvement in the results. As displayed in Figure 3.2, two algorithms A and B may achieve the same overall precision at a fixed high recall level in a given project. However, the candidate links list generated from algorithm B may
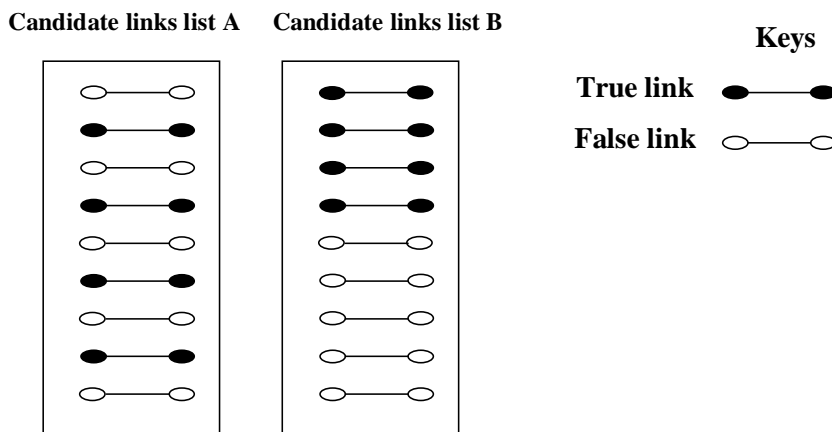
Figure 3.2 Two candidate links lists with the same overall precision but different precision on top of the list

contain more true links on the top, i.e. algorithm B obtains higher precision than A at lower recall levels. In this case measuring only the overall precision would fail to capture the improvement resulting from algorithm B.

To capture the change in the internal structure of the candidate links list, we also measure precision of retrieval at different threshold levels. More specifically, in our experiments precision is evaluated from low recall levels of 10%, 20% to the highest possible level at a 10% interval. The results at recall levels of 10% and 20% contain the links at top of the retrieval results that would be seen and inspected by the analyst first. Therefore the improved precision at these low recall levels is especially meaningful as it implies more true links are listed on the top of the retrieval results and the analyst may be able to evaluate those important links earlier in the process.

- **DiffAR**

DiffAR is a quantitative measurement proposed by Hayes et al. [35] to compare the difference between the average similarity scores of correctly retrieved and incorrectly retrieved links. It is computed as:

$$DiffAR = \frac{\sum_{(q,d)\in D_T} Sim(q,d)}{|D_T|} - \frac{\sum_{(q,d)\in D_F} Sim(q,d)}{|D_F|}$$

where *Sim(q,d)* represents the link similarity score computed by using one of the IR models, D_T is the set of candidate links that are true links, D_F is the set of candidate links that are false positives, and |D| is the number of the links in the set D.

A higher value of DiffAR indicates that true links have higher similarity

scores on average and therefore can be differentiated more easily from false positives in the retrieved results.

- **Lag**

Lag of a true link in the candidate links measures the number of false positives that are ranked higher than that true link. Lag of the whole candidate links list, i.e. the average Lag of all retrieved true links, specifies that on average how many false positives are found in the candidate links list above each of the true links. A lower Lag represents that true links are more separated from false positives.

- **Average Precision Change at various recall levels**

We are also presenting a new metric, the *Average Precision Change (AP)* at various recall levels, which provides a more accurate measurement of the precision changes in the retrieved links list than the overall precision at a specific recall level.

Let Δ_i (i=1,2,…,k) be the precision change after applying two different retrieval strategies at the i-th recall level for k different recall levels. The *Average Precision Change* at various recall levels is calculated as

$$AP = \sum_{i=1}^{k} \Delta_i / k \,.$$

In our experiments the precision changes Δ_i are evaluated at recall levels from 10% to the highest achievable recall with a 10% interval.

Previously proposed metrics such as DiffAR are not suitable for comparing different retrieval algorithms across different projects, as these metrics take values on different scales depending on the project. In contrast, the *Average Precision Change* takes values on a fixed scale and allows the evaluation of the impact of various retrieval algorithms across different datasets.

There are also other metrics that are used in general IR applications. The most commonly used ones include:

1) F-measure, a harmonic mean of recall and precision. As there is usually a trade-off between recall and precision, the F-measure is used to represent the balance between the two measures. The max of F-measure indicates the "best" achievable combination of precision and recall. F-measure can be computed as

$$F = \frac{(1+\beta^2) \times (recall + precision)}{recall + (1+\beta^2) \times precision}$$, where β is the weight used to tilt the balance between the two measures. When $\beta=1$, recall and precision are equally important. $\beta>1$ means recall is more important and vice versa.

2) Average Inverse Rank (AIR). A true link is assigned the inverse of its rank $(1/r)$ in the retrieved links list. AIR takes the average over the inverse rank of all true links in the list. A large value of AIR indicates that true links are ranked higher in the retrieval result.

Similar to the overall precision value at a specific recall level, F-measure is not sufficient to measure the improvement in the retrieval results. This measure would also fail to capture the change in the internal structure of the two retrieval results displayed in Figure 3.2. Therefore precision at various recall level is used in our experiments instead of the F-measure.

AIR also has its own limitations. Although an increased AIR can indicate 'good links rising, bad links sinking" in the candidate links list, and a decreased AIR reflects the opposite, this metric cannot measure the degree of the separation between the true links and false positives in the list. Therefore in our experiment, DiffAR is used to effectively assess the quantitative separation between the true links and false positives in the retrieved results.

3.3 Baseline Result

This section presents the results for the five datasets using the basic PN retrieval algorithm introduced in section 2.3. These results provide a baseline against which the enhancement strategies are compared. As the SE450 dataset contains fifteen projects, one representative project was chosen and the result in this project is shown in Figure 3.3. The results in all fifteen projects can be found in Appendix A.

As shown in Figure 3.3, the precision at high recall level of 80% and 90% in these datasets is relatively low, ranging from 3% in CM-1 to 36% in LC datasets. For instance, at a recall level of 90%, precision was achieved at 20%, 17% and 36% in IBS, EBT and LC respectively. (Note the chosen SE450 project can not achieve recall as high as 90% or 80% because a number of the true links in this project contain no shared terms and therefore are

non-retrievable by using term-based retrieval model)

We conducted exploratory analyses on the performance of the basic probabilistic retrieval algorithm to investigate the reasons for the low precision of the retrieved results in these datasets. The analyses have motivated the enhancement approaches presented in this thesis for improving the precision of the results. Chapters 4 through 6 will provide the details of those analyses of the baseline results.
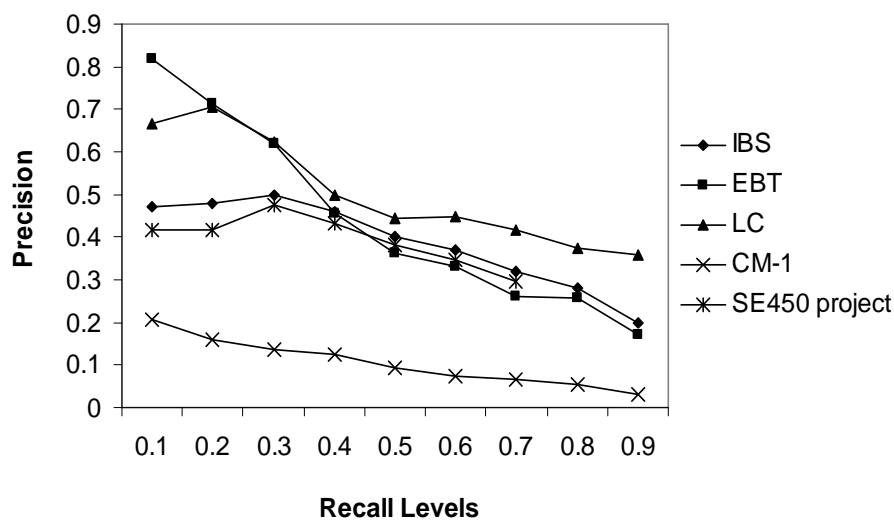
Figure 3.3: Baseline results on various datasets using the basic PN algorithm

CHAPTER 4: QUERY TERM COVERAGE AS AN ENHANCEMENT FACTOR

4.1 Introduction

The tracing tool Poirot implements the basic PN model and was previously evaluated against IBS, EBT and LC datasets. At a recall level of 90%, precision was achieved at 20%, 17% and 36% in IBS, EBT and LC respectively [9]. An exploratory analysis of the performance of the basic probabilistic retrieval algorithm on the IBS dataset was carried out to investigate the reasons for the low precision of the retrieved results. IBS was chosen for the analysis as it contains the largest number of requirements (164) and documents (71) among the three datasets. The investigation concentrated on two distinct sets of links that were incorrectly handled by the tracing tool. These were the top *false positives* (false links that are retrieved as correct) in the candidate links list and the top *false negatives* (true links that are falsely rejected) in the discarded links list. The recall of the retrieval algorithm can be directly improved by reducing false negatives, while precision can be directly improved by reducing false positives. On the other hand, when false negatives are pushed up in the ranking, some false positives among the candidate links can be rejected by 'raising the threshold' and therefore precision can also be indirectly improved.

Analysis of the top false positives showed that a large percentage of these pairs contained very few distinct matching terms and often just a single matching term that occurs multiple times. As an example, when the requirement "*A road section shall be added*" in IBS dataset is issued as the query, a class "*Tutorial GUI*" is returned by the tool simply because the only shared term "*section*" occurs frequently in this class. This is an example of an incorrect link as "*section*" refers to a tutorial section in the class diagram and represents a different and unrelated concept (related to a road) in the query.

In contrast, when examining false negatives we found that some of the incorrectly missed links contain a relatively high number of shared distinct terms. For instance, in the IBS system, a trace between the requirement "*Inventory shall be updated on receipt of a shipment*" and the UML class "*Material Inventory Database*" received a low probability score and was

incorrectly rejected by the tool because both shared terms "*inventory*" and "*update*" have very low weights according to the *tf-idf* weighting scheme.

An exploratory study was also conducted on IBS dataset to examine the number of shared terms for the true links and false links. We found that among 410 true links in this dataset, 59% of them (a total of 242) share two or more distinct terms, while among 2,394 false links, the percentage is only 25% (a total of 477), meaning the majority of the false links either share no common terms or one distinct term at most.

In the standard IR models, the similarity score between the query and the document is measured based on the weights of the shared terms regardless of how many distinct query terms co-occur in the document. This observation has led us to consider a more complex term weighting scheme that not only depends on the co-occurrence of individual terms but also gives more weight to groups of terms that are concurrent in both query and document.

Let's consider the following motivational example: assume we are tracing from a query $q=\{a,b,c\}$ that contains three single terms a, b and c to two documents. Document $d_1=\{a,b,c\}$ also contains a, b and c while document $d_2=\{a,a,a\}$ contains only a repeated three times. Assuming that the three terms $\{a,b,c\}$ have the same term weight, the basic IR models would generate the same similarity score for the two links, and would fail to capture the stronger relationship between q and d_1. Intuitively document d_1 should be considered more relevant to the query q, as q and d_1 share a larger number of distinct terms. Therefore their link should be assigned a higher similarity score.

To address this problem, we propose utilizing an enhancement factor named *query term coverage* as a contributing factor in the computation of the link probability score. Query term coverage represents the extent to which terms in the query are found in the traced document and is measured as the percentage of query terms occurring in the document. (Note that this metric measures the term coverage between individual query-document pairs. It is different from another metric named *query word usage* proposed in the thesis which is a measurement between a query and the whole document collection. The concept of query word usage will be discussed in section 4.4.2.)

For a link between query q and document d, query term coverage $TC(q,d)$ can be defined as follows:

$$TC(q,d) = \frac{m}{t} \qquad\qquad (4.1)$$

where t is the number of distinct terms contained in q, and m is the number of distinct matching terms found in both q and d. Low query term coverage occurs when very few distinct query terms are found in the document. The highest coverage value of 1 is reached when all query terms occur in the document.

An analysis was conducted to compare the potential effect of the TC approach on the retrieval results for the IBS project. The results are displayed in Table 4.1. Two tests were computed to compare the average query term coverage between true positives (correctly retrieved links), and false positives (incorrectly retrieved links). The first test compared the top 100 true positives with the top 100 false positives, and the second test compared all the false positives with all the true positives.

The tests showed that the average TC value of false positives is significantly lower than the average TC value for true positives at 5%

Table 4.1: Comparison on the query term coverage in false positives & true positives in IBS

Test	Group Type	# of links	Average query term coverage	Standard Deviation
1	**Top false positives**	100	0.298	0.121
	Top true positives	100	0.481	0.222
2	**All false positives**	1252	0.196	0.103
	All true positives	378	0.310	0.23

significance level. Thus the term coverage approach is more likely to increase the probability values for true positives, and to place more true links among the top retrieved links list, increasing the accuracy of the retrieval results. The improvement is expected to be more significant among the top-ranked retrieved links, indicating more true links may be returned first to the analyst for inspection.

Query term coverage has been investigated previously by researchers in the general IR context and also in the Natural Language Processing (NLP) field, but has never previously been investigated for purposes of trace retrieval. Rao et al. [59] considered the coverage factor in a TREC ad hoc task where document topic statements are supplied as the queries. In their experiments,

query term coverage was only applied to very short queries (containing only 2-3 terms) as the only factor that determines document relevance ranking. Their overall results did not show any improvement and they concluded that this was because the majority of the queries were relatively long and therefore were not affected by the factor at all. Burke et al. [5] also utilized coverage factor in their FAQ Finder system as one element to evaluate the similarity between the query question and the FAQ question in the database. Their task represented a more typical NLP problem and essentially compared sentences that expressed the same question using different terms and phrases. The incorporation of coverage metrics did not improve their retrieval results.

Although these applications do not support the use of query term coverage in information retrieval, the effectiveness of information retrieval methods has in general been shown to be rather context specific. Our initial investigation showed that the length of the requirements (queries) in our datasets is relatively short compared to the TREC queries. The average query length in IBS, EBT, LC and SE450 datasets is 5.7, 7.6, 9.6 and 9 terms respectively, while for the TREC collection queries the average length is longer than 20 terms [36, 37]. This observation implies that query term coverage could have a stronger effect on requirements trace retrieval than on TREC tasks. Furthermore, our investigation on the coverage factors among top ranked true and false links in IBS, as displayed in Table 4.1, indicates that the application of term coverage to automated requirements traceability can yield significantly better retrieval results.

The rest of this chapter will describe the TC factor (defined in formula 4.1) and how it can be incorporated into the PN retrieval model using three methods. The three methods of incorporating the TC factor are evaluated on IBS, EBT, LC, CM-1 and SE450 datasets, and the results are compared and discussed in section 4.3.

4.2 Incorporating Term Coverage into the Retrieval Algorithm

Extending the example in section 4.1, consider a query q containing t ($t>1$) distinct terms and two documents d_1 and d_2 where d_1 contains m ($1\leq m\leq t$) distinct matching terms with q while d_2 contains only one distinct matching term (m=1) with q but the term appears m times in d_2. Although d_1 has a higher query term coverage than d_2 and we believe $p(d_1/q)$ should be larger

than $p(d_2/q)$, since d_1 is conceptually more similar to the query q, the probabilities of the two links computed by the basic PN model will be equal, if all these matching terms have the same weight. We would like to increase $p(d_1/q)$ utilizing its high query term coverage of d_1 while maintaining $p(d_2/q)$ constant. One method towards this goal is to increase the basic probability of the links by an additive value associated with an enhancement factor $\frac{m-1}{t-1}$ which is a variation of the term coverage $TC(q,d)=\frac{m}{t}$. Note $\frac{m-1}{t-1}$ is equal to 0 for d_2 as d_2 contains only one matching term with q, therefore $p(d_2/q)$ will not be affected by this enhancement factor.

In our study we assume the enhancement degree to which $TC(q,d)$ affects the basic probability scores is proportional to the enhancement factor $\frac{m-1}{t-1}$. Three methods, defined as follows, were used to calculate the new probability score $p_c(d/q)$ that incorporates the coverage factor:

Method A : $\quad p_C(d\mid q) = [1+(\frac{m-1}{t-1})]\times p(d\mid q)$

Method B : $\quad p_C(d\mid q) = [1+2\times(\frac{m-1}{t-1})]\times p(d\mid q)$ $\hspace{2cm}$ (4.2)

$Method\,C: \quad p_C(d\mid q) = \begin{cases} m\times p(d\mid q) & if\ p(d\mid q)<1/m \\ 1 & if\ p(d\mid q)\geq1/m \end{cases}$

where $p(d/q)$ is the probability generated from the single term based on the probabilistic model introduced in section 2.3 and is calculated using formula 2.1.

In all three methods the enhanced probability score is linearly dependent on the query term coverage value $TC(q,d)$. The degree of the enhancement on the basic probability of the links increases when the coverage between the links becomes larger. The new probability score takes its maximum value (2 times as much as the basic score for method A and 3 times for method B) whenever all query terms are contained in the document, i.e. $t=m$.

In method C the new probability score is computed by multiplying the basic score by a factor equal to the number m of terms contained both in q and d. Thus this method provides the highest degree of enhancement among the three methods. Applying this method is motivated by our observation of the

link probability distribution of a few datasets in which we found the average probability score for candidate links is relatively low (around 0.01 in IBS and even lower in EBT and LC datasets). By using method C, true links are expected to rise toward the top of the candidate links list in a greater degree than method A and B therefore become more distinct from false positives.

4.3 Evaluation

In the empirical study, the three methods of incorporating the TC factor were evaluated on three datasets IBS, EBT and LC respectively. The best method was then chosen and further evaluated against the other two datasets CM-1 and SE450.

The conclusions from these experiments were:

- The TC method shows consistent improvement in precision when applied across almost all datasets compared to the basic PN model.

- Method C of incorporating the TC factor performed the best among the three.

- An analysis of the LC dataset in which the TC method decreased the precision among the top retrieved links suggests a potential improvement for this method by differentiating phrases and single words.

The experiments are described and discussed in detail in the rest of this section.

Evaluation of three TC methods on IBS, EBT and LC datasets

A good way to illustrate the global effect of the new algorithm is to display precision changes at various levels of target recall. Figure 4.1 shows the changes of the precision after applying the three TC approaches (labeled as TC_A, TC_B and TC_C in the graphs) in IBS, EBT and LC respectively at a range of target recalls from 10% to 90% with a 10% interval.

The results clearly show that the three TC approaches positively affect the precision of the retrieval results at most recall levels for the three datasets, and that method C of the new algorithm generally outperforms the other two methods except for low recall levels in the LC dataset.

As shown in Figure 4.1, at high recall levels of 80% and 90%, the precision of the retrieval in IBS and LC is considerably improved by an increase ranging from 2% to 9% for the three different methods incorporating the coverage factor, indicating that after applying the TC approach, more true links are retrieved and listed as the candidate links for the analyst. The improvement at low recall of 10% in IBS is especially significant with an increase of 46% (from 47% to 93%).

In addition, an evaluation on the datasets using the measurements DiffAR and Lag at recall level of 90%, as shown in Table 4.2, reveals that the TC approach positively changes the internal structure of the candidate links lists. Even for EBT which displays no change in the overall precision of the retrieval, both DiffAR and Lag are improved in the dataset. The value of DiffAR increases across all three datasets, indicating true links are more distinct from false positives in the candidate links list after applying the TC method. Similarly the value of Lag decreases after using the new algorithm, indicating the number of false positives above each true links decreases and therefore the two groups of links are more clearly separated from each other.

As summarized earlier, method C of the new algorithm outperforms the other two methods at high recall levels. However, LC experiences a decrease in the precision at 10%, 20% and 30% recall levels after applying the TC approach. An observation of the retrieval results in LC reveals that the probability scores of a number of false positives are incorrectly increased by a considerable degree. As a result these false links make their way into the top of the candidate links list. For example, a false link between a requirement "*The chosen light scene can be set by using the room control panel.*" and a class "*Admin Control Panel GUI*" is incorrectly enhanced by the new algorithm because they share three distinct terms "*room*", "*control*" and "*panel*". In fact, in this requirement "*room control panel*" is a phrase while in the class "*control panel*" refers to admin control panel and "*room*" appears in a different context.
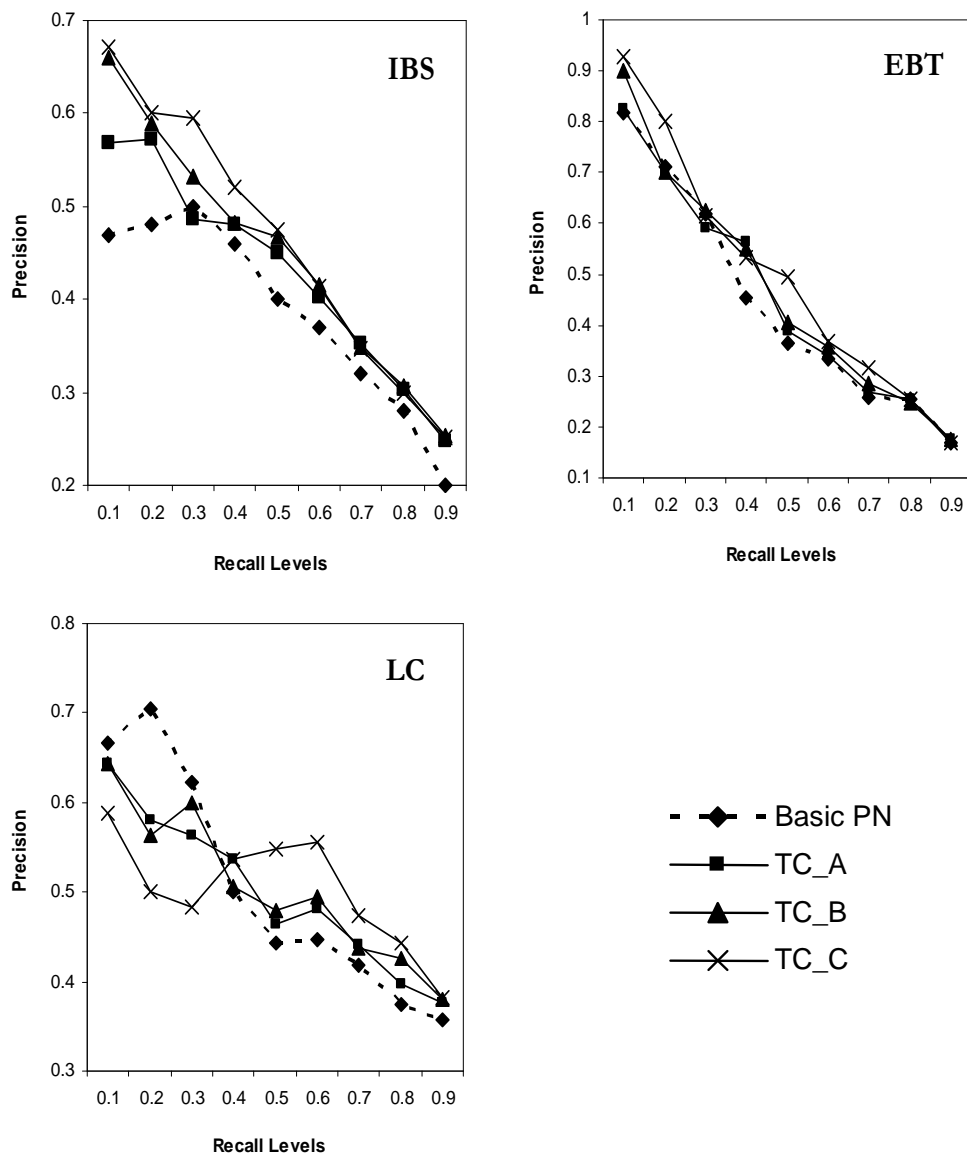
Figure 4.1: Precision values after three TC approaches at different recall levels in IBS, EBT and LC datasets

Table 4.2: DiffAR and Lag on three datasets at 90% recall

Dataset	DiffAR		Lag	
	Basic PN	TC*	Basic PN	TC*
IBS	0.03	0.10	254	222
EBT	0.01	0.05	188	120
LC	0.02	0.04	53	44

*Method C of the TC approach was used

This example reveals an important aspect through which the TC approach could be improved. All three shared terms are components of the phrase "*room control panel*" in the requirement. If the concept of query term coverage

differentiated terms contained in a phrase from single terms and considered phrase terms as a whole, the query term coverage between the requirement and the class in this example would be zero as "*room control panel*" does not co-occur in the class. Therefore the probability score of this false link would not have been enhanced.

The potential improvement on the TC approach is closely related to phrasing that is another enhancement strategy which was analyzed in my research and described in detail in section 5. The method of incorporating phrasing techniques into the TC approach is discussed in detail in chapter 5.

Evaluation on CM-1 and SE450 Dataset
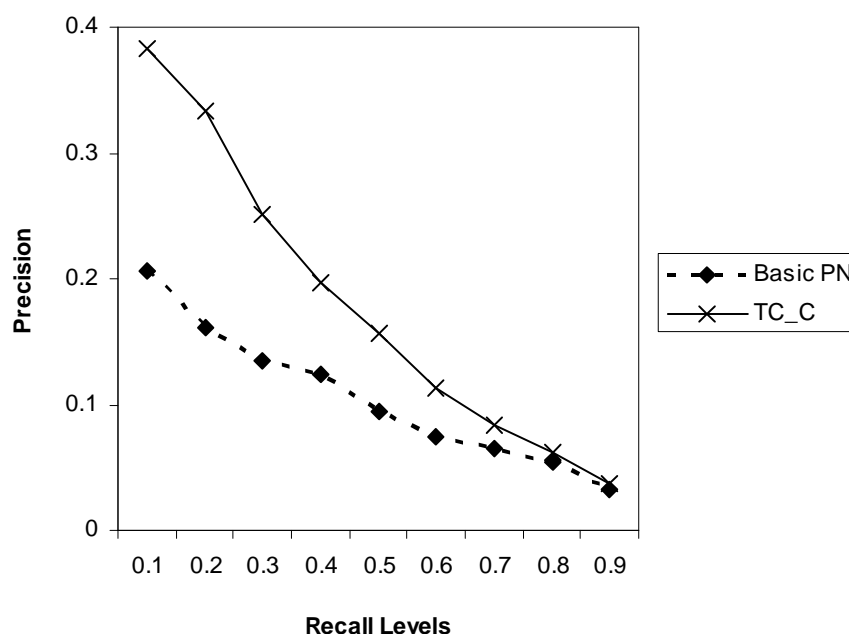
Figure 4.2: Precision change after TC at different recall levels for CM1

The TC approach is then evaluated against the large-scale dataset CM-1 and the 15 projects in SE450 dataset. CM-1 contains 235 high-level requirements and 220 low-level requirements, with a total of 361 true traces between the two types of documents. The tracing tasks on SE450 are to generate and retrieve links between a common set of 46 requirements and 15 different sets of java code of various sizes, ranging from 15 to 49 classes. Method C of the new approach is used as it consistently performed better in the previous experiments.

Similarly to the previous experiment, precision values at target recall ranging from 10% to 90% with a 10% interval are obtained. As shown in

Figure 4.2, the new algorithm integrating the TC factor consistently improves the precision at all recall levels for the CM-1 dataset compared to applying the basic PN algorithm. The improvement is especially significant among the top retrieved links for 10% and 20% recall levels, with an increase of 18% and 17% respectively, indicating a large number of true links have been pushed toward the top of the candidate links list.
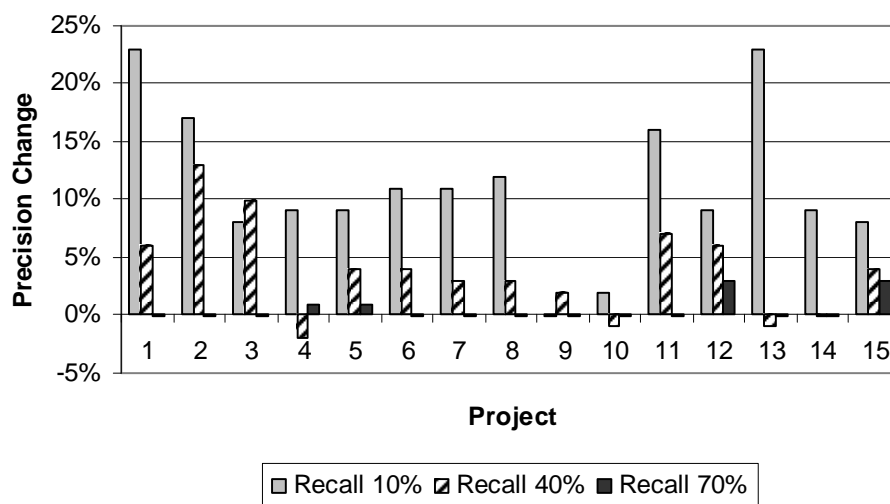
Figure 4.3: Precision change after TC in 15 projects at low, medium and high recall level

Figure 4.3 displays the precision change by using the TC approach at recall levels of 10% (low), 40% (medium) and 70% (high) in the 15 projects of SE450. (Note some of the projects can not achieve recall as high as 90% or 80% because a number of the true links contain no shared terms and therefore are not retrievable by using term-based retrieval model). The complete results for applying the TC method on all of the SE450 projects can be found in the appendix.

In the bar chart of Figure 4.3, the bars for 70% recall are missing for most projects (11 out of 15 projects), indicating no change at all in precision at high recall levels. Only four projects show a small change in precision at 70% recall. However, the change of the values of DiffAR and Lag for all projects, as displayed in Table 4.3, reveals that the TC approach had a positive impact in the structure of the candidate links list even though the overall precision remains the same. Our results show that DiffAR consistently increases and Lag decreases across all projects after using the new algorithm, indicating true

links had risen closer to the top of candidate links list than false positives.

Table 4.3: DiffAR and Lag on SE450 datasets at 70% recall

Project	DiffAR		Lag	
	Basic PN	**TC**	**Basic PN**	**TC**
1	0.009	0.031	75	67
2	0.034	0.073	78	60
3	0.003	0.035	103	88
4	0.028	0.055	125	116
5	0.007	0.031	174	145
6	0.016	0.047	87	71
7	0.0004	0.029	204	168
8	0.016	0.006	73	67
9	0.027	0.071	84	76
10	0.018	0.028	141	137
11	0.011	0.038	123	90
12	0.015	0.042	99	85
13	0.031	0.077	100	105
14	0.028	0.072	162	148
15	0.026	0.056	117	96

The change in the structure of the candidate link list could also be reflected by the change in precision at lower recall levels. As displayed in Figure 4.4, the precision at 10% and 20% recall levels is significantly improved for almost all projects, with an average increase of 10% and 11% respectively. At 10% recall the highest increase of 23% in precision is achieved on projects 1 and 13.
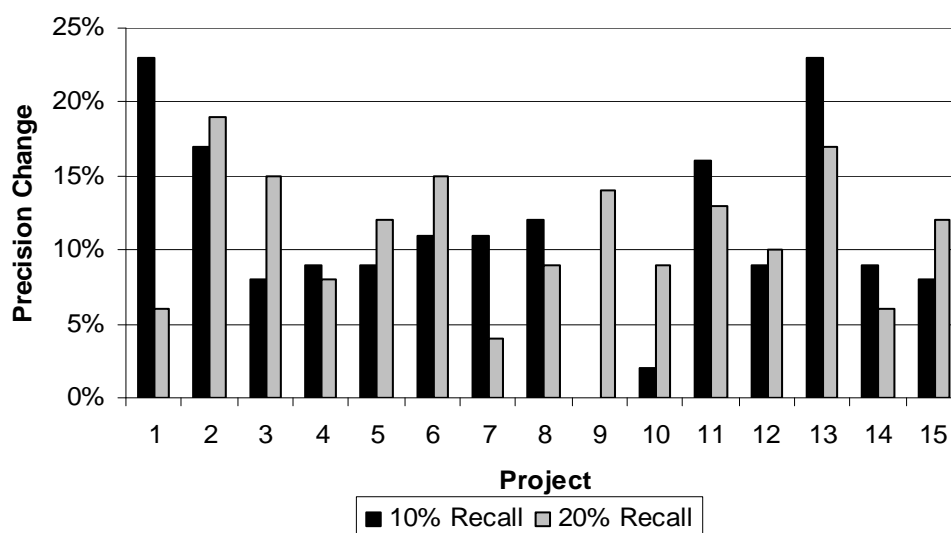
Figure 4.4: Precision change after TC in 15 projects at recall 10% and 20%

Project 9 demonstrates no change in precision at 10% recall. However, it gained an additional 14% of precision at a 20% recall level.

The evaluation results on the five datasets indicate that the TC approach has a positive effect in improving the retrieval results by increasing the overall precision, and furthermore generally pushing more true links towards the top of the candidate link list. Therefore the precision at the top of the candidate links list increases. Candidate links listed on top are returned to the analyst first. Therefore the analyst can evaluate many of the important links earlier. Furthermore, a higher precision on top of the candidate links list helps improve the analyst's trust in the tracing tool's accuracy in returning good links and therefore increases the tool's believability [35], a critical quality a tracing tool should exhibit.

CHAPTER 5: PHRASING IN AUTOMATED TRACE RETRIEVAL

5.1 Introduction

The use of phrases has been extensively investigated for content representation and document indexing in IR [6, 12, 23, 64]. In many traditional IR models, such as the VSM and PN models, documents are represented using single words as primitive elements. However single words are not necessarily ideal descriptors of document content. A single word can be associated with a wide range of concepts, which means that a model using single words as the primitive elements may retrieve many irrelevant documents and consequently reduce the precision of the retrieved results. Compared to single words, phrases, defined as a sequence of two or more words related through modification, are considered more accurate in specifying the document content. Applications that use phrases as part of the indexing language have been developed since the early days of IR research. Examples include phrase-based indexing in studies conducted by Cleverdon [10] and a series of experiments using phrase in the SMART IR system described by Salton [63].

Although the experimental results with phrases have been mixed [52], many studies have shown that use of phrases can yield improvement in the accuracy of document retrieval [6, 12, 23, 64]. Results from these studies have demonstrated substantial improvements obtained through the use of phrases in a broad variety of document collections. For example, Fagan et al [23] conducted experiments using five document collections and found the average increase of precision ranged from 2.2% to 22.7% when single word indexing was replaced by phrase indexing.

Our analysis in the trace retrieval results also indicates that using phrases may reduce the number of false positives and therefore improve the precision of the trace retrieval. Considering the example described earlier in Chapter 4 in which a false link between a requirement "*A road section shall be added*" and a class "*Tutorial GUI*" in IBS is retrieved by the basic PN model because of the single matching term "*section*", if phrase matching were used in addition to single term matching, phrases "*road section*" and "*tutorial section*"

would be a no-match. Other classes containing the matching phrase "*road section*" would have been assigned a higher relevance score and consequently, this false link could have been eliminated from the retrieval results by increasing the threshold. Such observations have motivated us to investigate the use of phrases in the requirements trace retrieval with a goal to improve the precision of the retrieval results.

5.1.1 Utilizing phrases in general IR field

Utilizing phrases in IR usually consists of three steps: 1) identify phrases in both queries and the document collections, 2) detect matching phrases between queries and documents, and 3) appropriately weigh phrases in the retrieval algorithm.

Step 1) Phrase Detection Methods

Commonly used phrase identification strategies fall into the two categories of "syntactical" and "statistical" approaches. Syntactical methods utilize the grammatical structure and relationships among words in a sentence to construct phrases, while statistical methods detect phrases based on the frequency and co-occurrence of terms. Both approaches have been implemented in IR applications to improve the accuracy of the retrieval algorithms. For example, Buckley et al. [4] integrated statistical phrases in their IR system SMART. In the system CLARIT developed by Evans et al. [22], syntactical phrases are extracted by using Natural Language Processing (NLP) techniques.

A typical statistical method requires defining several key parameters such as the maximum length of a phrase, the proximity of the occurrence of the phrase components, and document frequency threshold for both phrase and phrase elements [12, 23]. Fagan et al [23] conducted comprehensive studies on both statistical and syntactical methods for detecting phrases. The results suggest that the choice of the parameters in statistical methods greatly affects the retrieval accuracy, and that optimized parameters need to be defined for individual document collections. Another problem of statistical methods is that the quality of the detected phrases is often not satisfactory. Fagan's experiments showed that using statistical methods often result in the construction of inappropriate phrases due to some words that happen to occur in close proximity in the text, but that are not syntactically related. In the

meantime, some good phrase descriptions are missed because their component words are not in a close proximity [23].

Compared to statistical methods, syntactical approaches are considered more accurate in identifying phrases [23]. Two approaches of syntactical phrasing methods, template-based and parser-based approaches, have been frequently used. Template-based approaches match word groups within a document against a template library that defines the parts of speech expected to occur within a phrase (for example 'noun-preposition-noun'). The FASIT (Fully Automated Syntactically based Indexing) system developed by Dillon et al. [16] is a typical application of this approach. In contrast, parser-based methods attempt to analyze entire sentences or significant parts of them in order to produce syntactic phrases. Examples of the applications utilizing parser-based method include the work conducted by Fagan et al. [23] in which a PLNLP (Programming Language for Natural Language Processing) parser was used to produce a complete parse of the document text.

Step 2) Phrase Matching

Detecting the matching phrases in the queries and the documents is the next step for using phrases in IR after phrases are identified. The contiguous components of a phrase in a query are not necessarily adjacent and may be several words apart when the phrase is used within a document text. Some phrasing techniques [31] therefore have taken into consideration the proximity, or lexical distance, between phrase components in the searched documents. However, the studies conducted by Pickens and Croft [52] suggest that adjacent phrases, phrases whose constituent words are next to each other in the document text, are considered better indicators of document relevance than conjunct phrases, phrases whose constituent words appear at a certain lexical distance in the document.

Step 3) Phrase Weighting

Weighting of the detected phrases is another issue that needs to be addressed before using phrases in IR applications. Similarly to single words, phrase weighting techniques include *tf*, the phrase frequency within one document, and *idf*, the inverse document frequency of phrase occurrence within the whole collection. However, after exploring various term weighting schemes Salton and Buckley [67] concluded that judicious weighting of words

is preferable to any weighting of phrases. Other techniques for phrase weighting include lexical distance between phrase components. For example, Hawking and Thistlewaite [31] proposed to weight phrasal terms proportional to their proximity. Studies on phrase weighting methods however show that none of the schemes have significantly better performance than others [49].

5.1.2 Phrasing in Requirements Tracing

Despite the extensive work on phrasing in the IR area, we believe this is the first study that applies an automated phrasing approach to improve the IR tools for automated requirements traceability. Hayes et al. [33, 35] investigated retrieval with key-phrases in which a VSM model was extended by treating a set of technical terms or key phrases as single terms. Their approach assumed that there are definition and/or acronyms sections in the requirement documents from which the key phrases can be easily extracted. However, the lack of such information in many existing software projects limits the applicability of their approach. The phrasing strategy presented in this thesis will solve this problem by automatically detecting and reconstructing phrases from across all software artifacts in the system.

To apply phrasing to the trace retrieval problem, there are three issues that must be addressed:

1) Selection of the phrase categories:
   The phrase categories that should be detected for the purpose of requirements tracing should be selected. For example, should we consider noun phrases, adjective phrases, or more categories?

2) Phrase detection method:
   A method needs to be defined for actually detecting phrases within the set of traceable documents.

3) Phrase incorporation:
   Phrasing concepts need to be incorporated into the existing trace retrieval algorithm.

The three issues will be discussed in detail in the following sections.

The rest of this chapter will therefore introduce a phrasing strategy by addressing each of the three important issues individually. This phrasing strategy is evaluated against IBS, EBT, LC, CM-1 and SE450 datasets, and the results are discussed in section 5.5. The synergistic integration of phrasing and

the TC approach is also introduced and evaluated for all datasets. This chapter also discusses a study that extends the basic phrasing strategy by considering alternative phrase categories and longer phrases.

5.2 Phrase Categories and Phrase Detection Method

Most phrasing methods have been focused on noun phrases [52] which are composed of a head noun and its modifier such as an adjective or another noun, as noun phrases capture the most concrete content of a sentence. Many studies of general IR applications have suggested nominal compounds, a sequence of two or more nouns related through modifications, to be the most useful type of noun phrases to improve the retrieval accuracy [26]. Our small-scale manual analysis of software artifacts such as requirements specifications, comments in source code, and text embedded in design diagrams also indicates that nominal compounds occur more frequently than other types of phrases such as adjective-noun phrases. Examples include argument and method names in java classes or UML diagrams, which typically are composed of a combination of verbs and two-word nominal compounds.

Our study of phrasing in trace retrieval therefore is focused on two-word nominal compounds. Section 5.6 describes a study on the extension of the phrase categories to include other types such as adjective-noun phrases which also occur in requirement documents, and nominal compounds with more than two words. Short phrases are usually preferred over long complex phrases for phrase-based indexing in IR, because they have a better chance of matching short phrases contained in the query and can still match the long phrases based on the short phrases they have in common. For our analysis, the length of an acceptable nominal compound was established at two words. Two forms of nominal compounds were considered in our experiments. The first was a noun-noun phrase such as "*weather forecast*", while the second was a noun modified by a prepositional noun, such as "*condition of road*", which can be identified and re-constructed as the phrase "*road condition*".

In a typical IR environment, text is normally comprised of relatively complete sentences. Requirements that typically consist of one or more complete sentences can be easily parsed by the part-of-speech (POS) tagger, and the appropriate tags are produced in a relatively reliable fashion. However many types of software engineering artifacts, such as method and class names

in UML and code, tend to be very succinct and contain only a few words. In these circumstances the incomplete syntactic information makes it difficult for a POS tagger to correctly identify the syntactic components. As an example, a POS tagger would falsely analyze a Java class method named "*set road sensor*" as three contiguous nouns. Without the context information, it is difficult to identify the word "*set*" as a verb or a noun.

This problem is solved by recognizing that a phrase can only impact the similarity score between a requirement and a traceable document if it occurs in both of them. Phrases appearing on only one side have no ability to impact the tracing process. It is therefore possible to initially detect candidate phrases by using the POS tagger to search through the textually rich requirements and then simply check to see if the candidate phrases are present in the set of traceable documents. For the purposes of this work a phrase is therefore defined as a "*noun phrase that appears in both the requirement and the document to be traced*".

A freely available parser-based POS tagger named 'Qtag' is used to identify phrases [76]. Qtag uses a dictionary to identify the syntactic category of each token in the text and outputs a series of POS tags that represent grammatical classes of words, such as nouns, verbs and adjectives etc, for each token in the input text. In addition to Qtag's high accuracy in phrase detection, it is also easily incorporated into a traceability tool and provides significant flexibility for detecting a variety of phrase types.

As an illustration of the phrase identification, consider the Qtag output of the POS tags for the following requirement:

The	*system*	*shall*	*provide*	*a*	*summary*	*report*	*of*	*weather*	*conditions*	
DT	NN	MD	VB	DT	NN	NN	IN	NN	NNS	
over	*a*	*specified*	*period*	*of*	*time.*					
IN	DT	JJ	NN	IN	NN					

where NN represents a noun and NNS represents a plural noun. Table 5.1 provides a complete description of the tag types. Notice that a component such as "*report of weather*" where "*report*" is modified by prepositional noun "*weather*" is also identified as a noun phrase. The resulting candidate phrases extracted from this requirement are: *summary report, weather report, weather*

conditions, and *time period*.

Table 5.1: Part-of-Speech tag sets

Tag	Description	Tag	Description
DT	Determiner (a, this, that, the)	IN	Preposition (on, of)
NN	Noun, common singular	NNS	Noun, common plurals
MD	Modal auxiliary (may, will)	JJ	Adjective, general (near)
VB	Verb, base		

5.3 Phrase Incorporation

This section presents an enhancement of the basic probabilistic retrieval algorithm designed to improve its retrieval precision through considering phrases that co-occur in both queries and traceable documents. If both query *q* and document *d* contain the same phrase, this provides a compelling indication of the relevance of the document with respect to the query and suggests that they should be linked.

Each query *q* is parsed using the phrase identification algorithm described in section 5.2 to identify phrases. Let S_{PH} be the set of terms contained in the phrases found in the query. If no phrases are found, S_{PH} is empty. Traces between the query *q* and a document *d* are evaluated by computing a conditional probability value $p_{PH}(d/q)$ representing the relevance of document *d* in respect to query *q* as follows:

Phrasing algorithm T:

$$p_{PH}(d \mid q) = p(d \mid q) + p_f(d \mid q) \tag{5.1}$$

where $p_f(d|q) = \dfrac{\sum_{t_i \in S_{PH}} p_f(d|t_i) p_f(q,t_i)}{p(q)}$

The first component *p(d/q)* is the basic probability value computed in expression (2.3) defined in chapter 2 using only the single terms co-occurring in *d* and *q*. The second component $p_f(d/q)$ represents the contribution to the probability value provided by phrasing. The factors $p_f(d/t_i)$ and $p_f(q,t_i)$ are computed similarly to the analogous terms in the basic probability value, but they depend only on the frequency of terms contained in phrases. If a term t_i appears in a phrase, its contribution to the overall probability $p_{PH}(d/q)$ is

increased by two times. Notice that the probability value $p_{PH}(d/q)$ is equal to the basic term-based probability score if the query and the document have no phrases in common.

An alternative method of incorporating phrases into the retrieval algorithm was also explored in our study as phrasing algorithm PH, and is defined as:

$$p_{PH}(d|q)=p(d|q)+p_f(d|q) \tag{5.2}$$

Unlike the algorithm T described in formula (5.1) in which a detected phrase between the query-document pair contributes to the similarity score based on the *tf-idf* weight of its constituent single terms, in this alternative algorithm the phrase contribution to the similarity score is the function of the *tf-idf* weight of the phrase as a whole. Therefore the second component is calculated as $p_f(d|q) = \dfrac{\sum\limits_{f_i \in S_{PH}} p_f(d|f_i)p_f(q,f_i)}{p(q)}$. This component represents the contribution of phrases to the basic probabilistic value. Factor $p_f(d/f_i)$ in this component is computed as $p_f(d|f_i) = \dfrac{freq(d,f_i)}{\sum\limits_{k} freq(d,t_k)}$, where $freq(d,f_i)$ is the frequency of phrase f_i in the document d. Note in this component the phrase frequency is normalized by the total occurrence of all single terms in the document rather than the occurrence of phrases only. As phrases usually occur much less frequently than single terms, this normalization schema is used to assure that factor $p_f(d/f_i)$ is within a reasonable scale. The other two factors are calculated as $p_f(q,f_i) = \dfrac{freq(q,f_i)}{n_i}$ and $p(q) = \sum\limits_{i} p(q,t_i)$, where n_i is the number of documents that contain phrase f_i.

Similarly to the basic probabilistic retrieval tool, probability scores $p_{PH}(d/q)$ for any given query *q* are computed for all documents in the searchable document collection *{d₁, d₂, …, dₙ}*, using one of the above two phrasing algorithms. Potential traces with query *q* are established for those documents whose probability score falls above a certain threshold.

5.4 Synergistic Incorporation of Phrasing and TC

A natural extension to two enhancement approaches introduced so far is to investigate a synergistic incorporation of both TC and phrasing into the

probabilistic retrieval model. The incorporation of both factors can be implemented by applying phrasing followed by query term coverage. Therefore the new enhanced probability $P_{TCPH}(d/q)$ between a query q and a document d that incorporates both approaches was calculated similarly to formula (4.2), using one of the three methods for defining the coverage effect except that $p(d/q)$ is replaced by $p_{PH}(d/q)$. The expression is defined as follows:

Method A : $\quad p_{TCPH}(d \mid q) = [1 + (\dfrac{m-1}{t-1})] \times p_{PH}(d \mid q)$

Method B : $\quad p_{TCPH}(d \mid q) = [1 + 2 \times (\dfrac{m-1}{t-1})] \times p_{PH}(d \mid q)$

(5.3)

Method C : $\quad p_{TCPH}(d \mid q) = \begin{cases} m \times p_{PH}(d \mid q) & if\ \ p_{PH}(d \mid q) < 1/m \\ 1 & if\ \ p_{PH}(d \mid q) \geq 1/m \end{cases}$

where $p_{PH}(d/q)$ is the enhanced probability determined by applying phrasing using either of the algorithms introduced in section 5.3.

5.5 Evaluation

To assess the effectiveness of the phrasing strategy, the two phrasing algorithms, referred to as phrasing algorithm T (based on phrasal terms) and PH (based on whole phrases) respectively in the experiment, were firstly evaluated against IBS, EBT and LC datasets. The results show that the phrasing algorithms based on the phrasal terms performed better than the alternative algorithm. The new algorithm was then further evaluated against the CM-1 and SE450 datasets. In section 5.5.2, a second evaluation describes the experiment of applying the synergistic approach that incorporates both TC and phrasing algorithms on all five datasets.

The conclusions from the evaluation are:

- Phrasing algorithm T generally performs better than algorithm PH;
- The phrasing approach effectively improves the precision at top of the retrieval results for most of the datasets, but has insignificant effect on precision at high recall levels;
- The phrasing approach does not guarantee improvement on precision for all datasets. The effectiveness of this approach may depend on

certain characteristics of the given project.

- Among all methods, the synergistic application of TC and phrasing (using method C of TC, and phrasing algorithm T) together is shown to be the most effective in significantly improving precision at 10% and 20% recall levels for almost all datasets.

The experiments are described and discussed in detail in the rest of this section.

5.5.1 Evaluating two phrasing algorithms in IBS, EBT and LC

Figure 5.1 displays the effect of applying phrasing to retrieve traces for three datasets at recall levels from 10% to 90% with an interval of 10%. As shown in the graphs of Figure 5.1, both phrasing algorithms (Phrasing_T and Phrasing_PH) improve the precision in IBS and LC datasets at high 80% and 90% recall level, while EBT had no change in the overall precision. However an evaluation of DiffAR on the retrieval results in EBT shows an increase on DiffAR (from 0.03 to 0.05) after applying both phrasing algorithms, indicating that true links have been enhanced more significantly than false positives and therefore have become more separated from them.

Compared to algorithm PH, phrasing algorithm T performs relatively better with higher increase in precision for IBS and LC at almost all recall levels. For EBT phrasing algorithm PH yields the lowest precision compared to the basic PN model and the phrasing algorithm T.

As clearly shown in the graphs in Figure 5.1, precision in LC is consistently improved across all recall levels when algorithm T is applied. The most significant improvement is achieved at a recall level of 80% for this dataset, with an increase of 10%. For IBS, the improvement is less significant than LC, but it also shows an increased precision at almost all recall levels. Compared to LC and IBS, EBT is the least responsive dataset toward the phrasing algorithm. The precision is improved only at middle recall levels on this dataset, with a very small increase. Based on these results, phrasing algorithm T is selected as the strategy for incorporating phrases.
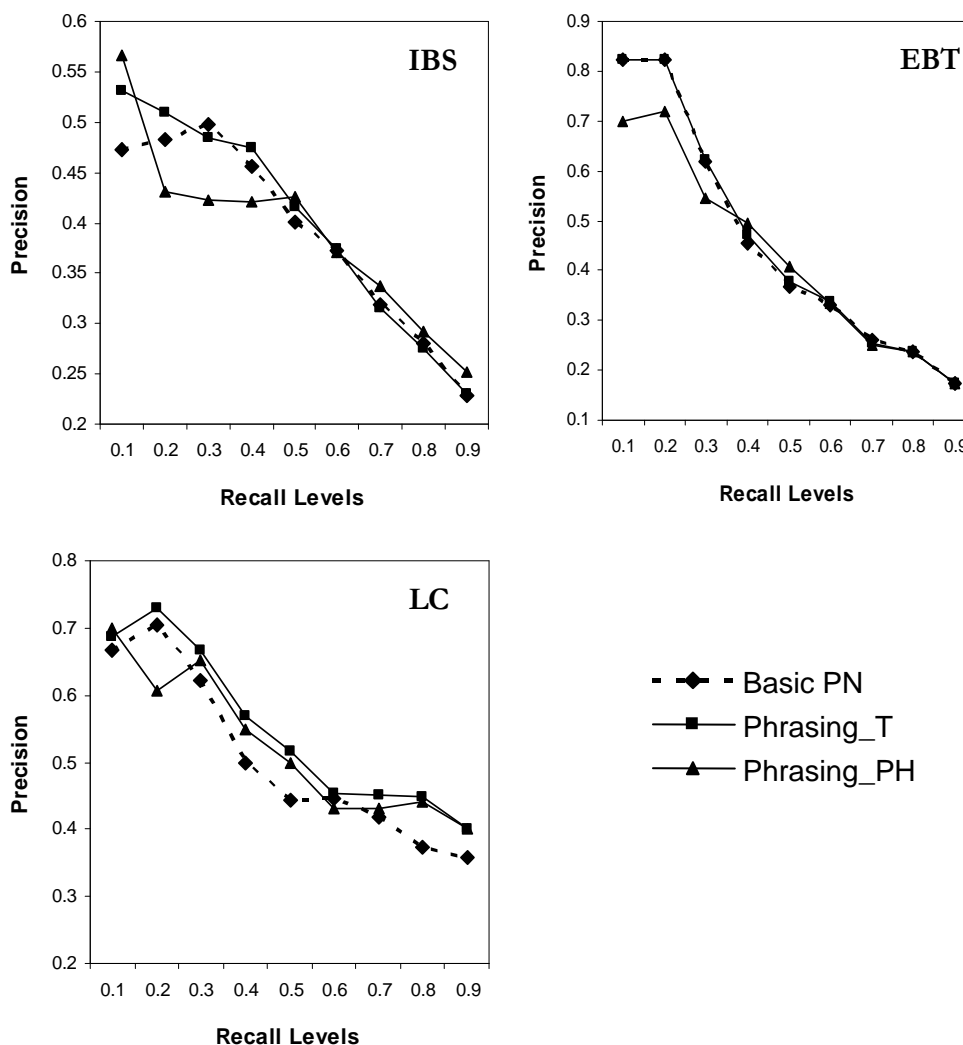
**Figure 5.1: Effect of phrasing algorithms on three datasets
at different recall levels**

By examining the true links missed by phrasing in all the three datasets, we observe that there are very few or even no common terms in the missed query-document pairs, and that most of those single terms do not belong to a phrase. This observation explains why phrasing has negligible effect at high recall level of 90%. As phrasing will only enhance the probability between a query and a document when at least one shared phrase exists between them, the probability scores for most of these missed true links were equal to the basic term-based probability value.

The results overall illustrate the positive effect of phrasing for improving the retrieval results. With more good links being pushed towards the top of retrieved links, the tracing tool is able to raise the threshold to reduce the

number of false positives in the candidate links list while retrieving the same number of true links. As a result the precision of the retrieval results can be improved.

5.5.2 Evaluation of phrasing algorithm T on CM-1 and SE450
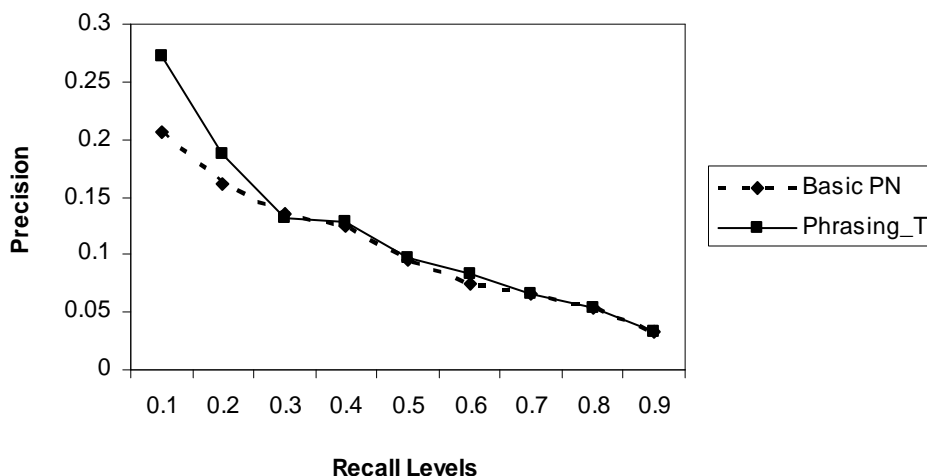
Figure 5.2: Effect of phrasing algorithm on CM-1

The phrasing algorithm T is then evaluated against the CM-1 and the 15 projects of the SE450 dataset. Figure 5.2 displays the recall versus precision scatter-plot at various recall levels for the CM-1 dataset by applying the basic PN retrieval algorithm and the phrasing approach (algorithm T). The phrasing approach consistently improves the precision in this dataset at almost all recall levels with the most significant improvement of 7% occurring at a recall level of 10%, indicating more true links have been included at the top of the candidate links list and returned to the analyst earlier.

The evaluation of the results for all projects in the SE450 dataset shows that there is no change in precision at high recall levels after phrasing is applied. Similarly to the analysis of the previous phrasing results for IBS, EBT and LC datasets, the true links that are missed by the basic PN algorithm are between document pairs that share a very small number of terms and no phrases at all. Therefore after applying the phrasing approach the probability values of these links remain the same.
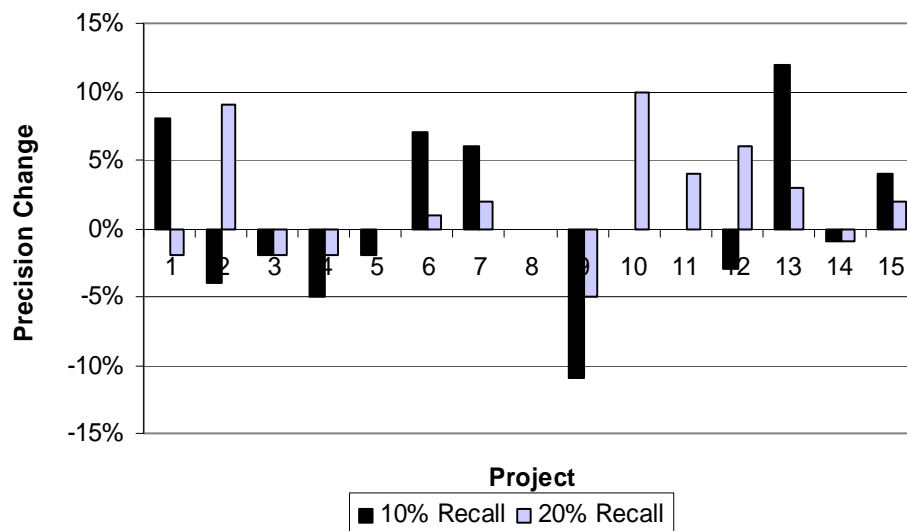
Figure 5.3: Precision change after phrasing in 15 SE450 projects

However, when considering the precision at a low recall level, i.e. precision at the top of the candidate links list, the retrieval results with the phrasing approach yield changes at various degrees, as displayed in Figure 5.3. The graph in Figure 5.3 displays bars representing the change in precision at 10% or 20% recall levels after using the phrasing approach. The bar heights vary significantly, suggesting that phrasing has an inconsistent retrieval performance on the 15 projects. Despite the observed improved performance in a few projects in which precision at both recall levels is increased, some projects experience a decrease in precision at least at one recall level. Project 9 shows the worst precision change, with a decrease of 11% at recall of 10% and 5% at recall of 20%. By observing the results of project 9 we find there are a few false links incorrectly enhanced by phrasing. Examples include a false link between a long requirement which specifies how to decelerate when approaching an obstacle and a class named "*Maximum Speed*". The probability of the requirement-document pair was increased because of the shared phrase "*maximum speed*" between them. However this is in fact a false link as the requirement is only referencing maximum speed in order to calculate the deceleration speed of a vehicle. The two artifacts are not directly related and therefore the probability value between them should not have been enhanced.

The example implies that phrasing does not guarantee improvement for any given software project. The effectiveness of phrasing may be influenced

by certain characteristics of a specific project. The work on identifying and investigating such factors will be discussed in Chapter 7 of this thesis.

5.5.3 Evaluation of the synergistic approach

Table 5.2: Precision comparison on enhanced algorithms at recall of 90% in IBS

Method		precision
Basic PN		20%
Basic PN +TC	Method A	25.1%
	Method B	26.6%
	Method C	28.9%
Basic PN + Phrasing		21.0%
Basic PN +TC +Phrasing	Method A	25.0%
	Method B	27.1%
	Method C	**28.9%**

The incorporation of both factors was firstly evaluated against IBS to compare the effect of the three different methods of incorporating the TC factor along with phrasing. IBS was chosen for the analysis as it contains the largest number of requirements and documents among the three smaller datasets. The comparison between applying the new synergistic approach and one enhancement strategy only is displayed in Table 5.2. The results indicate Method C of the synergistic approach achieved the highest precision of 28.9% at recall level of 90% among all the methods. It appears that applying both phrasing and coverage methods did not achieve additional improvement on the overall precision. However, an examination of the top candidate links shows that the precision among these links was significantly higher after phrasing and term coverage approaches were applied to the retrieval algorithm, as displayed in Table 5.3. The precision at recall level of 10% increased by 25% when both approaches were used compared to the precision achieved by the algorithm incorporating only term coverage. We therefore applied the

Table 5.3: Precision-on-top comparison for enhanced algorithms in IBS

Method	Precision at recall of 10%
Basic PN	47.2%
Basic PN + Coverage	68.3%
Basic PN + Coverage + Phrasing	**93.5%**

*Results were from using method C of query term coverage approach.

synergistic approach by incorporating the TC using the method C and phrasing using the algorithm T in the evaluation.

Evaluation on IBS, EBT, LC and CM-1 datasets

The effect of the synergistic approach on the precision at various recall levels for IBS, EBT, LC and CM1 datasets is illustrated in Figure 5.4 by comparing the precisions achieved by using the basic PN, phrasing and the TC approach individually. The graphs in this figure show that applying the new enhancement algorithm (labeled as "TC+PH" in the graphs) consistently improves precision at almost all recall levels for the IBS dataset, except at 90% recall level where the synergistic method achieved the same precision as the TC approach. The result implies that the new algorithm outperformed the other two enhancement methods in improving the retrieval precision in the IBS dataset. Compared to the basic PN, the increase ranges from 8% among most retrieved links (at 90% recall level) to 46% among the top retrieved links (at 10% recall level) after applying the new algorithm. Similarly to IBS, the EBT and CM-1 datasets also observed improvement of precision at some recall levels. The increase of precision is generally less significant than in IBS. The improvement is because after applying phrasing and the TC approach together in these datasets (IBS, EBT and CM-1), many false links that were assigned with high ranks by the basic PN model have sunk down in the candidate links list, while many true links have risen towards the top of the candidate links list. This change helps the analyst separate false links from true links in the candidate links list more easily.

As explained in section 4.4, a few false links in LC are incorrectly enhanced by the TC algorithm because they share three distinct terms "*room*", "*control*" and "*panel*" but in fact "*room control panel*" is a phrase in the requirement while in the class "*control panel*" refers to admin control panel and "*room*" appears in a different context. As a result, the LC dataset experiences a decrease in the precision at recall levels of 10%, 20% and 30% after applying the TC approach. The deterioration of the precision at top of the candidate links list on LC is mitigated after applying phrasing together with the TC approach. The hybrid approach increases the precision at 10% recall by a considerable 13% compared to using the phrasing method. However, at recall level of 20% and 30%, the precision is still lower than the basic PN

algorithm.
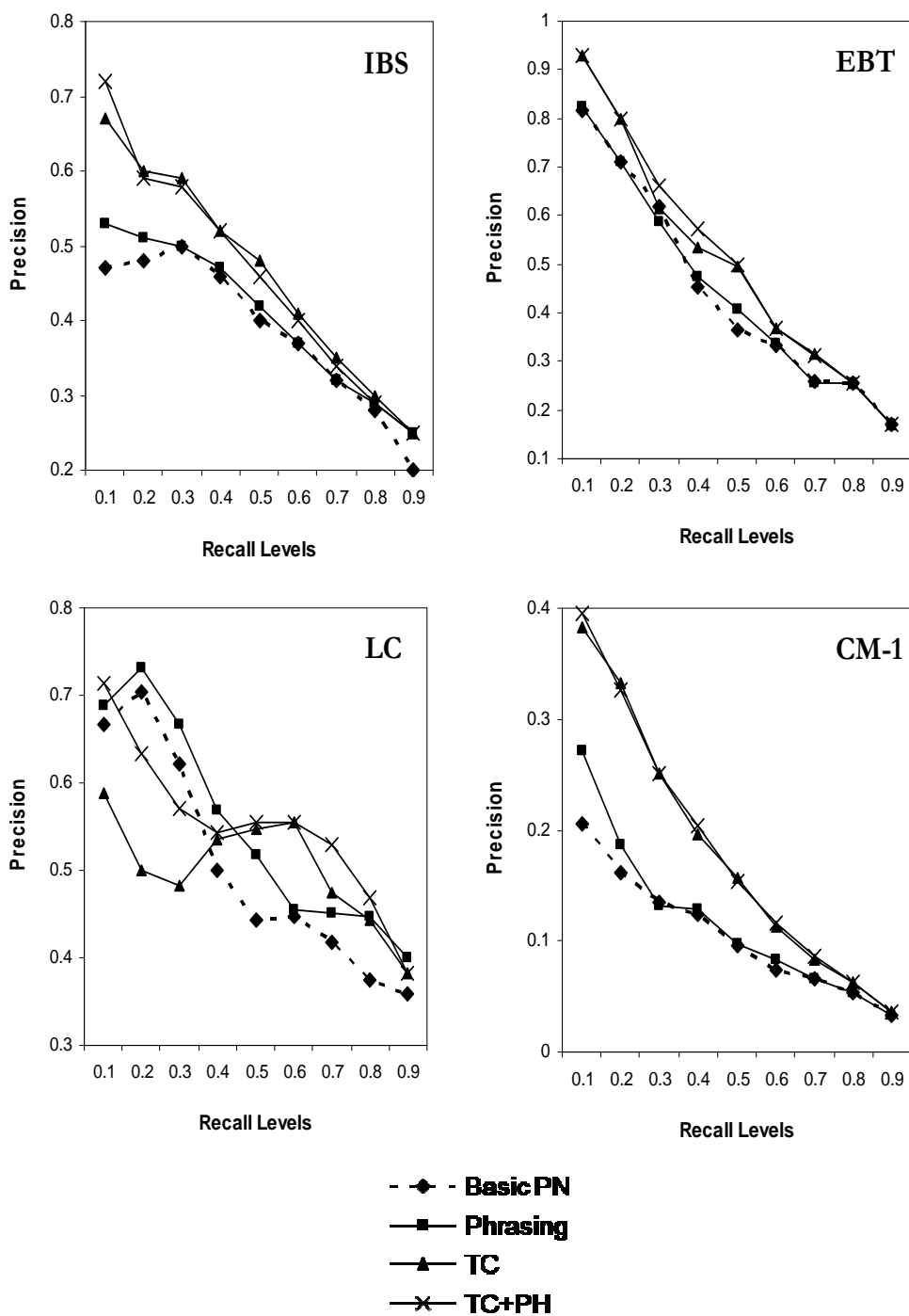
Figure 5.4: Evaluation results of TC, phrasing and the synergistic approach in IBS, EBT, LC and CM-1 datasets

Evaluation on SE450

The synergistic approach is also applied to the 15 projects in the SE450 dataset. For the purpose of demonstration, one representative project is chosen

from the 15 projects in the SE450. The results for this project are presented in Figure 5.5. (The entire results for all the projects in SE450 can be found in Appendix A.) Both phrasing and TC are able to consistently improve the precision at most of the recall levels for this project, especially at 10% recall level where the precision is improved by 17% and 21% individually. The method incorporating both TC and phrasing also improves the precision considerably. The improvement is especially significant at 10% recall where the precision increases to 91%. A positive impact on the precision is also observed in all fourteen other projects in the SE450 after the synergistic approach is applied, with the precision improvement at most of the recall levels, as recorded in Appendix A.

5.6 Studies on Extending Phrasing Approach

5.6.1 Phrasing with adjective-noun phrases

Our analysis of the software artifacts including requirement specifications and design documents suggest that adjective-noun phrases containing two words also occur in the document text. An exploratory analysis of the datasets is then conducted to examine whether or not considering adjective phrases in the phrasing algorithm may help improve the retrieval results.

The study is conducted firstly on IBS, EBT and LC datasets to examine
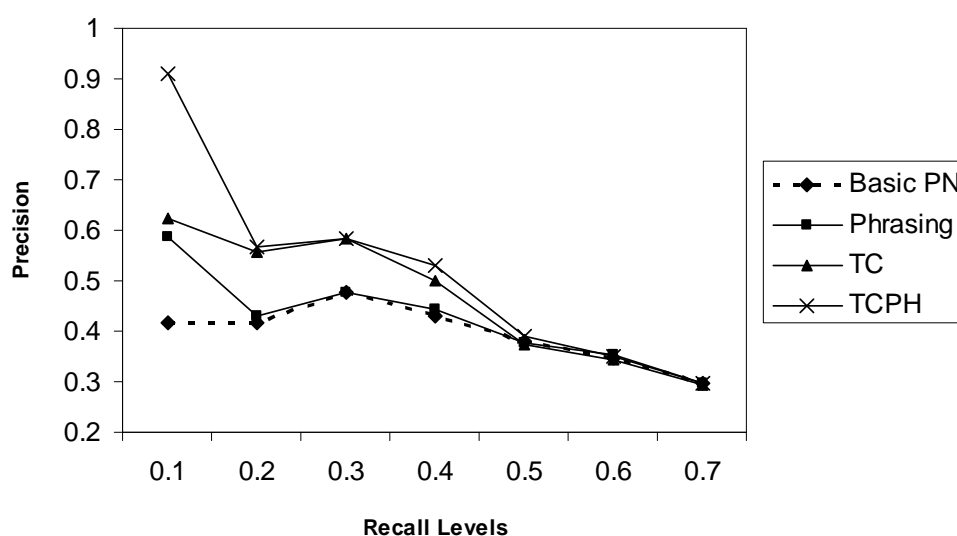
Figure 5.5: Effect of the synergistic approach incorporating both TC and phrasing in one project of SE450

whether adjective phrases occur frequently in the requirements and the traced documents. The results reveal that adjective phrases are used much less frequently in the documents than noun-noun phrases. For example, in the IBS dataset, noun phrases occur 89 times in the requirements and 35 times in the document, while for adjective-noun phrases the numbers are 47 and 5 respectively. The difference between the frequencies of noun phrases and adjective phrases is even greater for LC dataset, in which noun phrases are used 18 times in the requirements and 5 times in the document, while for adjective phrases the numbers are only 7 and 1 time respectively. The observation suggests that considering adjective-noun phrases in the proposed phrasing approach may not be able to achieve significant improvement in precision.

The conclusion is confirmed in an experiment in which the POS tagger is set such that two-word adjective-noun phrases are also detected and considered in the retrieval algorithm. The new retrieval algorithm incorporates the phrasing approach using equation (5.1) (algorithm T) and is applied to IBS, EBT, LC and CM-1 datasets.

Compared to the approach considering only noun-noun phrases, there is almost no difference in precision after the new phrasing approach is applied in these datasets. The only exception is in the EBT dataset, as displayed in Figure 5.6 in which the extended phrasing algorithm is labeled as *Phrasing _Adj* and the original algorithm considering only nouns as *Phrasing_NN*. The precision at recall of 50% for this dataset experiences a small decrease of 3%. The retrieval of false links at this recall level in EBT is caused by an adjective-noun phrase "*real time*" which happens to occur in a requirement and some UML classes that are not related to this requirement.

The study suggests that considering adjective-noun phrases in the phrasing algorithm is not necessarily useful in improving the precision of the trace retrieval. (Note this study is exploratory and the observation is based on the datasets available to us.) The phrasing approach should only focus on noun-noun phrases in order to achieve the best trace retrieval results.

5.6.2 Phrasing with three-noun phrases

Our previous experiments on phrasing are focused on phrases of two words. The example described in section 5.5 where two three-word phrases

"*room control panel*" and "*admin control panel*" are mistakenly considered a match suggests that extending phrase length from two-word to three-word to
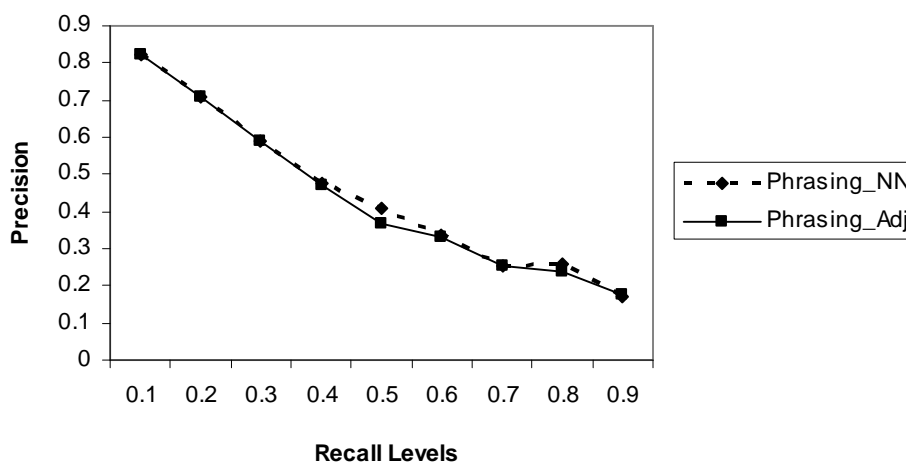
**Figure 5.6: Effect of the extended phrasing algorithm
with Adjective nouns on EBT**

prevent such mismatching may improve the retrieval results.

A study is then conducted on all the available datasets to investigate whether three-noun phrases are more useful than their sub-phrases (two-noun phrases) in improving the precision of the trace retrieval. The study, however, suggests that the attempt to avoid mismatches by considering the matches on the entire long phrases can decrease the precision. Examples include a true link in IBS between a requirement "*An alert shall be issued for maintenance scheduling time*" and a UML class "*maintenance schedule*" in which the probability value would not be improved if the match were strictly limited to the entire phrase, as the three-noun phrase "*maintenance scheduling time*" in the requirement and "*maintenance schedule*" that occur in the class would be considered a no-match.

Such situations in which the phrasing algorithm based on long phrases would worsen the result outnumbered the situation in which the new phrasing approach would be beneficial, indicating that the three-word phrases are not as useful as two-word phrases for improving the trace retrieval results. The conclusion is confirmed by the experiment of applying the phrasing approach based on three-word noun phrases to the available datasets. As shown in Figure 5.7, the results from this experiment show that after the new approach is applied, the precision is mostly lower than the results from the phrasing
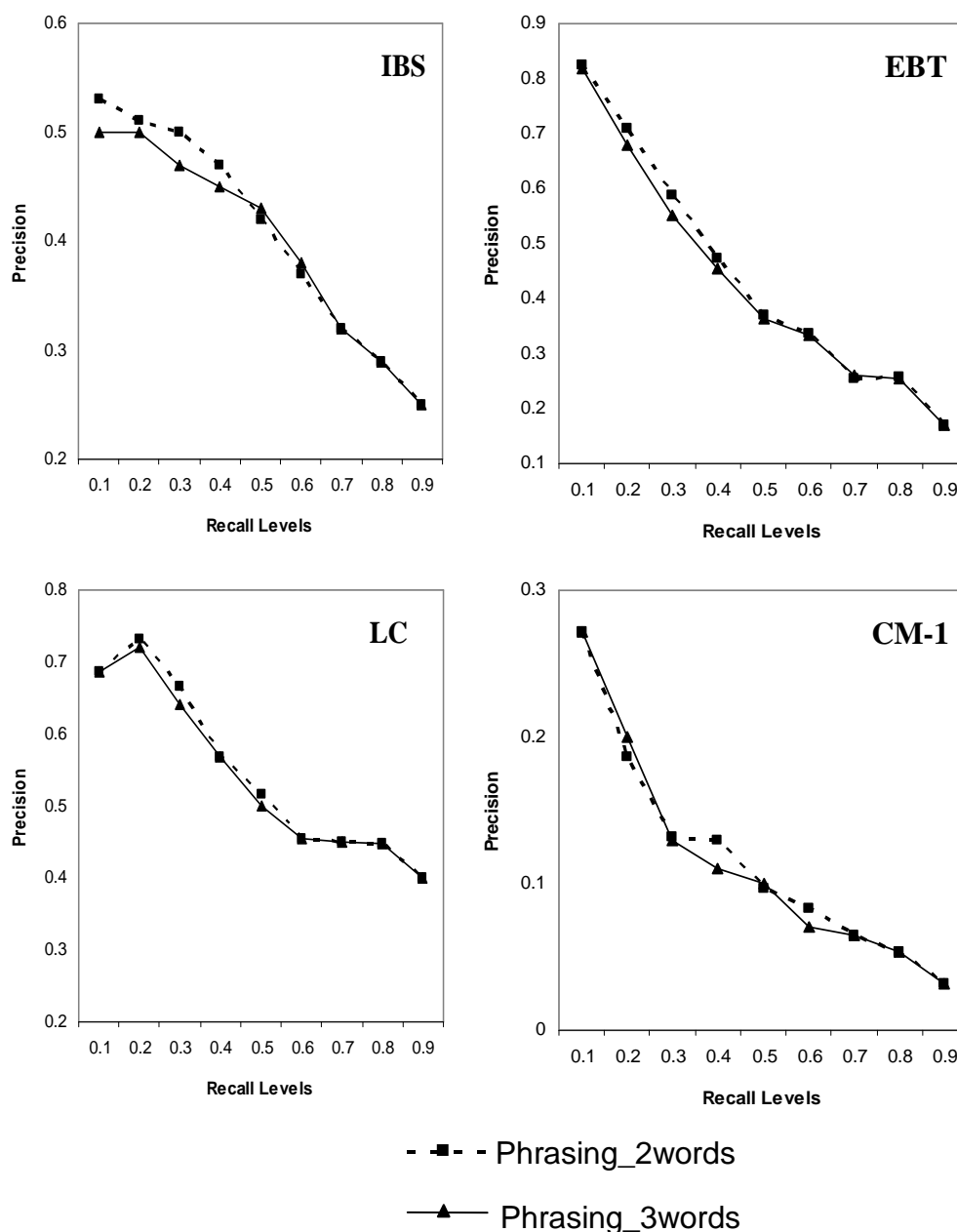
**Figure 5.7: Evaluation results of phrasing with 3-word nouns
in IBS, EBT, LC and CM-1 datasets**

based on two-noun phrases in almost all recall levels for these datasets. (The results for this experiment on the fifteen projects in SE450 are presented in detail in Appendix A.)

Hence, the studies of extending the phrasing approach suggest that the phrasing algorithm based on phrases containing two nouns is the most efficient in improving the trace retrieval results.

CHAPTER 6: UTILIZING PROJECT GLOSSARY DATA TO IMPROVE RETRIEVAL RESULTS

6.1 Introduction

One concern about the phrasing approach described in Chapter 5 is that it does not take into account the quality of the detected phrases. All phrases are assumed to be equally meaningful and to contribute equally to strengthening the belief in a link. However, there are usually certain phrases or terms in a project that are more significant for capturing the critical meaning of the project. If such phrases or terms are found in both the query and the document, they should provide a stronger contribution to the probability score between the two artifacts.

As an example, consider the following requirement belonging to the IBS system that was previously discussed in section 5.2: "*The system shall provide a summary report of weather conditions over a specified period of time*". Four phrases were identified by the automatic phrase detection method: *summary report, weather report, weather conditions* and *time period*. Based on our understanding of the IBS system it would be intuitive to assign higher weights to phrases such as *weather report* and *weather conditions* rather than to *summary report* and *time period*.

As many systems utilize project glossaries to define such important terms and phrases, the glossary can be used to identify terms that should be weighted more heavily than others. The glossary is typically defined early in the project and analysts and developers are encouraged to utilize the defined terms within the software requirements specifications, design documents, and other artifacts. In addition to adding importance to certain phrases and terms, the project glossary may also contain additional phrases that the syntactic parser is unable to discover. These phrases may for example not fit into the prescribed grammatical template, or may contain more than the standard number of terms.

This chapter introduces a new retrieval algorithm that assigns higher weights to key terms and phrases defined in a project glossary. The synergistic approach that incorporates TC, phrasing and the glossary method is also described in section 6.3. The evaluation of these approaches is described and

discussed in section 6.4.

6.2 Applying Project Glossary with Phrasing

Let $S_{PG}=\{k_1,k_2,...,k_m\}$ be the set of keywords identified in the project glossary. The set $S_{PH}=\{t_1,t_2,...,t_n\}$ contains the terms in the phrases detected using the POS tagger that are not in the project glossary. The new probability $p_{PG}(d/q)$ of a link between a document d and a query q is computed using both the information in the project glossary and the phrases detected by the phrasing approach. The probability $p_{PG}(d/q)$ is defined as follows:

$$p_{PG}(d \mid q) \propto p(d \mid q) + p_f(d \mid q) +$$

$$\delta \sum_{k_i \in S_{PG}} \frac{p(d \mid k_i)p(q \mid k_i)}{p(q)} + \delta \sum_{t_i \in S_{PH}} \frac{p_f(d \mid t_i)p_f(q \mid t_i)}{p(q)} \qquad (6.1)$$

where δ is a multiplicative factor ($\delta>0$) that augments the weights of the terms and phrases that appear in the project glossary.

In the expression (6.1) for $p_{PG}(d/q)$ we assume that the contribution of including information from phrasing and the project glossary is additive. If no terms or phrases defined in the project glossary appear concurrently in a document-query pair, then the probability of a link between the pair is the same as the one computed using the simpler $p_f(d/q)$ formula defined in expression (5.1) for the phrasing approach. The formula for $p_{PG}(d/q)$ assigns a higher weight to keywords and phrases that are defined in the project glossary, and their contribution to the overall probability $p_{PG}(d/q)$ depends on the chosen $\delta \geq 0$. Phrases that are detected by the tagger but are not included in the project glossary contribute to the overall probability at the same level defined in expression (5.1) in chapter 5.

Notice that in expression (6.1) $p_{PG}(d/q)$ is defined to be proportional to the right-hand side of (4.6) which may take values larger than one. Values for $p_{PG}(d/q)$ that obey standard probability constraints (i.e. values vary in [0,1] and sum up to one for all traces to q) can be computed after a simple rescaling. However as the ranking of the $p_{PG}(d/q)$ values is not affected by the rescaling, no further transformation of $p_{PG}(d/q)$ is required.

6.3 Synergistic Incorporation of TC, Phrasing and Glossary

Intuitively the three enhancement methods *TC*, *Phrasing* and *Project Glossary* can be implemented synergistically by applying first the project glossary along with phrasing, and then the TC approach. Experiments that are

not reported in this thesis are also conducted to evaluate the performance of applying the three enhancement methods in different orders (see Appendix B). The synergistic method introduced in this thesis consistently outperforms the other approaches on the available datasets. The enhanced probability $P_{TCPG}(d|q)$ between a query q and a document d that incorporates all three approaches is calculated similarly to expression (4.2) for basic term coverage, except that $p(d|q)$ is replaced by the new probability score $p_{PG}(d|q)$ defined in expression (4.6). The expression is defined as follows:

$$p_{TCPG}(d\,|\,q) = \begin{cases} m \times p_{PG}(d\,|\,q) & if\ \ p_{PG}(d\,|\,q) < 1/m \\ 1 & if\ \ p_{PG}(d\,|\,q) \geq 1/m \end{cases} \qquad (6.2)$$

6.4 Evaluation

The effectiveness of utilizing a project glossary could only be evaluated for the IBS and SE450 projects as no glossary was available for the other datasets. The project glossary of the IBS contained 34 entries composed of 6 keywords and 28 phrases. For the SE450, its project glossary defines 4 keywords and 6 phrases. In the experiment the impact of different δ values on the retrieval results were first evaluated. The glossary approach and the synergistic approach that incorporates all three enhancement methods were then evaluated.

The conclusions of these experiments are:

- δ value of 0.5 appears to be optimal for incorporating the glossary approach;
- The retrieval performance of the project glossary approach varies significantly from dataset to dataset and seems to depend on the extent to which a glossary is consistently utilized during the development of software artifacts.
- When a meaningful project glossary is available, the synergistic application of the TC, phrasing and project glossary methods yield the highest increase in precision among the top ranked retrieved links.
- The impact of the synergistic approaches that incorporate multiple enhancement methods seems to be closely related to the effectiveness of individual enhancement methods.

The experiments are described and discussed in detail in the following section.

6.4.1 Evaluation of the glossary approach and the synergistic approach

Four different values for δ are evaluated in an experiment using the IBS dataset. Table 6.1 shows the effect of the project glossary approach on the retrieval performance using four different δ values equal to 0.2, 0.5, 0.8 and 1.0. Although different δ values do not show any significant effect on the overall retrieval accuracy, as displayed in Table 6.1, the δ value of 0.5 achieves the highest precision of 74% among the top retrieved links (10% recall levels). Hence, the value of $\delta=0.5$ is chosen in the following experiments.

Table 6.1: Effects of δ values on precision in IBS data using the project glossary approach

Recall	Precision			
	$\delta=0.2$	$\delta=0.5$	$\delta=0.8$	$\delta=1.0$
90%	24%	**25%**	25%	25%
10%	73%	**74%**	73%	69%

The project glossary approach and the synergistic approach incorporating all three enhancement methods are evaluated against the IBS and the fifteen projects in the SE450 datasets. The effect of each approach on the precision for IBS and one selected representative SE450 dataset is illustrated in Figure 6.1, which compares the precision values achieved for the basic PN against all other proposed enhancement approaches. The full results for all the projects in SE450 can be found in Appendix A.

The results indicate that the effect of applying the glossary approach vary from dataset to dataset. For the IBS dataset, the precision at 10% and 20% recall levels increases significantly, indicating more true links have risen to the top of the candidate links list after integrating the phrasing approach with project glossary, as displayed in the graph on the left in Figure 6.2. Compared to the basic PN, the new glossary-based approach applied to the IBS dataset obtains a significant precision increase of 17% (from 47% to 64%) at recall level of 10%. The synergistic approach incorporating the TC, phrasing and the glossary approaches achieves an additional increases of 2% in precision compared to utilizing glossary and phrasing only, indicating more true links
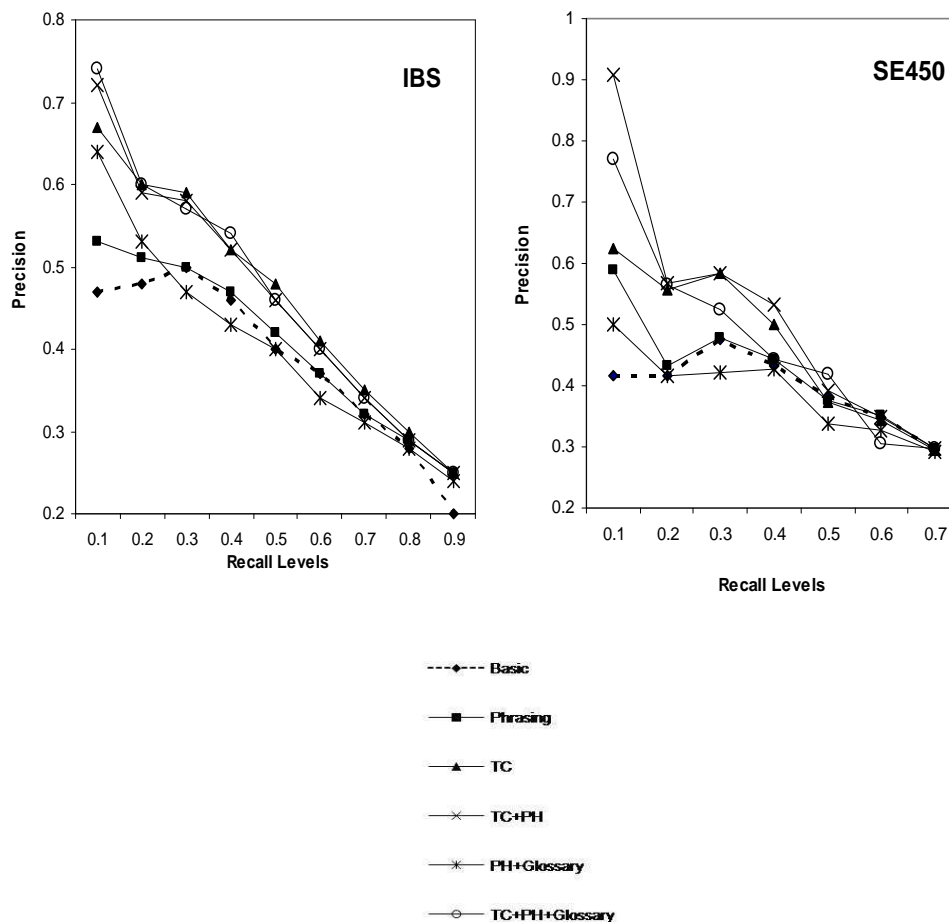
Figure 6.1: Results of utilizing a project glossary in
IBS and one selected project in SE450

are included in top ranked candidate links. However no additional improvement is obtained for IBS on the overall precision when recall is equal to 90% after the glossary approach is applied. This can be explained because no project terms or phrases are found in the links with very low probability values.

By examining the 28 phrases detected in the project glossary of IBS, we found that 24 of them were among the 126 phrases previously detected by the POS tagger. This suggests that the additional increase in precision achieved by using the project glossary was primarily from the enhanced weighting assigned to the phrases and key terms contained in the project glossary.

In contrast, the precision at 10% and 20% recall levels decreases up to 16% for the SE450 project. The decrease in precision at low recall levels is observed for most of the projects in SE450 (see Appendix A for more details). The analysis of the traces retrieved for the SE450 projects reveals that the

project glossary approach assigns high relevance scores to irrelevant links between unrelated pairs of requirements and Java classes because of the co-occurrence of weak glossary terms, such as "*vehicle*", that were inconsistently used in the two documents collections to indicate different concepts. Such incorrect traces appear among the top retrieved links and cause the decrease in precision for low recall levels. This observation suggests that the traceability of a software system may not be enhanced if a project glossary is not consistently utilized during the development of software artifacts. The work on evaluating the usefulness of an available project glossary will be described in Chapter 7.

6.5 Summary of all Three Enhancement Methods

Chapters 4, 5 and 6 have compared the performance of the three term-based enhancement traceability methods at various recall levels. The results show that in general the proposed enhancement approaches are effective in improving the precision of the retrieval results compared to the basic algorithm, but that the extent of the improvement differs from project to project.

Among all methods, the synergistic application of TC and phrasing together is shown to be the most effective and significantly improves precision of the top retrieved links corresponding to 10% and 20% recall levels for almost all datasets. When a meaningful project glossary is available as in the IBS dataset, the synergistic application of the TC, phrasing and project glossary methods yield the highest increase in precision among the top ranked retrieved links. The impact of the synergistic approaches that incorporate multiple enhancement methods seems to be closely related to the effectiveness of individual enhancement methods. This is evidenced in particular in the SE450 dataset, where the project glossary approach has a negative effect on the precision of the retrieval results.

The TC method (algorithm T) shows consistent improvement in precision when applied across all datasets compared to the basic PN model. For example in IBS the precision at 10% recall increased significantly by 20% after using the TC method, and at other recall levels the increase in precision was also substantial ranging from 2% to 12% (as displayed in Figure 4.1). The only exception was observed in the LC dataset (see Figure 4.7), where the TC

method improved the overall precision at higher recall levels (from 40% to 90% recall), but lower precision was observed among the top retrieved links (10%, 20% and 30% recall). As discussed in section 4.3, the problem for the LC dataset was caused by the occurrence of longer phrases containing terms that could be used individually to refer to different concepts. In fact when phrasing is applied with TC, this problem is partially addressed and the precision among the top retrieved links (10% recall level) increased significantly.

The phrasing algorithm achieved considerable improvement in precision for IBS, LC and CM1. The improvement resulting from phrasing is generally less significant than the TC method. In the EBT dataset, the effect of using phrasing was almost unnoticeable. Some projects in the SE450 dataset even experienced a decrease in precision when phrasing was applied. Further details about the results for other SE450 datasets are reported in Appendix A.

Similarly the effect of applying the glossary approach has not been consistent. The analysis of the traces retrieved for the SE450 projects in which the project glossary approach was not effective revealed that the decrease in precision was caused by some glossary items being used inconsistently in the documents collections.

These results have shown that the enhancement methods may be more effective for certain datasets than for other ones. This observation motivated the following research questions: "*Is there a set of characteristics in the individual projects that impacts the effectiveness of these enhancement approaches?*" In other words, "*Can we predict whether an individual approach will be effective in a given project prior to running any retrieval algorithm?*"

This research question has motivated the work presented in the next chapter investigating characteristics of software projects that can explain differences in performance of the enhancement strategies. The study also develops metrics to predict the effectiveness of the enhancement strategies for a specific project.

CHAPTER 7: PREDICTORS FOR ENHANCEMENT STRATEGIES

Document characteristics may vary greatly from one project to another. For example, documents may have different sizes, types, lengths, and use different vocabulary. Such differences can affect the performance of the various retrieval approaches. Our study has identified a set of metrics and dataset characteristics as possible predictors for the effectiveness of the enhancement approaches. The predictors can be employed to identify which enhancement algorithm should be used in the tracing tool to improve the retrieval performance for specific documents collections.

The first section of this chapter will then introduce two predictors, namely *average Query Term Coverage* (*QTC)* and *average Phrasal Term Coverage* (*PTC)*, for the TC and the phrasing methods respectively. These two predictors are evaluated against all five available datasets and the experiments are described in section 7.2. Furthermore, an iterative technique is described in section 7.3 that determines which trace retrieval approach is more effective on a new project when no prior knowledge of traces is available, and builds a partial answer set containing information collected from user feedback. This approach enables an enhancement strategy to be turned on or off according to feedback provided by the user. As part of the previous effort of developing the predictors, this chapter also describes two factors, *query length* and *query word usage*, that had been previously investigated for predicting the effectiveness of the TC method.

7.1 QTC and PTC as Predictors for TC and Phrasing Methods

An intuitive metric for the term coverage approach is developed using the Query Term Coverage defined in section 4.1. For instance, for the IBS dataset, the TC approach is very effective and achieves consistently higher precision than the basic algorithm. The study presented in section 4.1 also shows that the correctly retrieved links in the IBS project have higher Query Term Coverage on average than the incorrectly retrieved traces (the comparison is shown again in the bottom half of Table 7.1 for reference). Thus the TC approach is more effective if queries have higher Query Term Coverage.

The same study is conducted against the LC dataset where the TC approach performs poorly among the top retrieved links list. As shown in the top half of Table 7.1, the analysis of the Query Term Coverage values for the top 50 retrieved links in the LC dataset reveals that the correct links have lower average Query Term Coverage values than the top 50 incorrectly retrieved traces. The analysis explains the reason that the precision at top of the candidate links list in LC decreases after the TC approach is applied.

A project-level predictor for the effectiveness of the TC method for a given project p is computed as the *Average Query Term Coverage* of p for all queries q_i and documents d_j. The predictor denoted by $QTC(p)$ is defined as follows:

$$QTC(p) = \frac{\sum_i \sum_j TC(q_i, d_j)/n_j}{n_q} \qquad (7.1)$$

where *TC (q_i, d_j)* is the *Query Term Coverage* value defined in expression (4.1), n_j and n_q are the total numbers of documents and queries in project p, respectively. Only document/query pairs that share two or more distinct terms are considered in the above calculation, as the TC method has no impact on pairs that have only one term in common.

In a similar fashion a function of the co-occurrence of phrases in queries and traceable documents can be defined as a metric for predicting the effectiveness of the phrasing approach in a given project. *Phrasal Term*

Table 7.1: Comparison on the average query term coverage in false positives & true positives in LC and IBS

Test	Group Type	# of links	Average query term coverage	Standard Deviation
LC dataset				
1	Top false positives	50	0.297	0.089
	Top true positives	50	0.283	0.101
2	All false positives	350	0.192	0.056
	All true positives	80	0.273	0.009
IBS dataset				
1	Top false positives	100	0.298	0.121
	Top true positives	100	0.481	0.222
2	All false positives	1252	0.196	0.103
	Top false positives	100	0.298	0.121

Coverage for a query *q* and a document *d* is defined as $PC(q,d)=\frac{m}{t_q}$ where *m* is

the number of terms in phrases that are shared by *q* and *d*, and t_q is the total number of distinct phrasal terms in query *q*.

In terms of the phrasing approach, *Phrasal Term Coverage* emphasizes the extent to which the phrases contained in a query are used in a traceable document. Therefore the maximum value for the *Phrasal Term Coverage* of a query-document pair is equal to 1 when all phrases used in the query are found in the document.

The A*verage Phrasal Term Coverage* of a project *p*, denoted as *PTC(p)*, is defined as the average Phrasal Term Coverage *PC (q_i, d_j)* for all queries q_i and documents d_j and is computed as follows:

$$PTC(p)=\frac{\sum_i\sum_j PC(q_i,d_j)/n_j}{n_q}$$
(7.2)

where n_j and n_q are the total numbers of documents and queries in the project, respectively. Thus if a project has high *Average Phrasal Term Coverage* value, then the phrasing algorithm is expected to retrieve a higher proportion of correct links and therefore increase the precision of the retrieval results.

To some degree, our work on the predictors for the enhancement retrieval approaches is relevant to the study of query performance in the general IR context. The fact that users often issue poorly-performing queries that retrieve a large number of irrelevant documents has led IR researchers to investigate the degree of ambiguity of a query with respect to the collection of document being searched. Cronen-Townsend et al. proposed in [13] to measure the degree of dissimilarity between the language usage associated with the query and the generic language of the collection as a whole. Their proposed method, *clarity score*, is the Kullback-Leibler divergence [11] calculated as a smoothed function of the relative frequency of terms in both the query and the whole document collection. Although their results indicated the clarity scores might be used to predict the poorly-performing queries, this approach is often criticized because of the time-consuming computation involved in calculating the probability distribution of all single terms in the entire collection. Therefore the applicability of *clarity score* to the general IR applications is

strongly undermined.

Similarly, He and Ounis [36, 37] proposed a *simplified query clarity score* which simplifies the computation of the distribution of individual terms in clarity score. Their results also indicated the strong correlation between their proposed metric and the query performance.

Unlike *clarity score* that is developed to measure the performance of individual queries, the concepts *average Query Term Coverage* (*QTC)* and *average Phrasal Term Coverage* (*PTC)* defined in this section are designed to be used as project-level metrics that measure the performance of the TC and phrasing methods in a entire project. More specifically, these two measures represent the degree in which words or phrases contained in a query co-occur in the documents the query searches against. Thus *QTC and PTC* attempt to provide information on the potential performance of the TC and the phrasing approaches respectively for a given project, while *clarity score* focuses on the performance of the IR model on the document collection with regard to a specific query. The values of the two measures for existing requirements can be obtained offline prior to the retrieval. Additionally, as *QTC* and *PTC* do not take into account the distribution of query terms in the entire document collection, the computation of the measures for free-form queries is still very simple and efficient.

7.2 Evaluating Predictors for TC and Phrasing Methods

The *Average Query Term Coverage* and the *Average Phrasal Term Coverage* metrics defined in the previous section were computed for each of the available datasets: IBS, EBT, LC, CM-1 and the fifteen projects in SE450, a total of nineteen projects. The performance of both the TC and the phrasing methods for each project were compared against the basic PN algorithm by measuring the average precision change at various recall levels achieved by the enhancement retrieval methods.

The association between the *QTC* and the precision improvement by applying the TC method is depicted in Figure 7.1(a), where the points correspond to the nineteen projects used in the analysis. The x-axis and the y-axis represent the *QTC* values and the average precision change achieved by the TC approach, respectively. The scatterplot in Figure 7.1(a) shows a strong positive association between the two variables, indicating that higher average
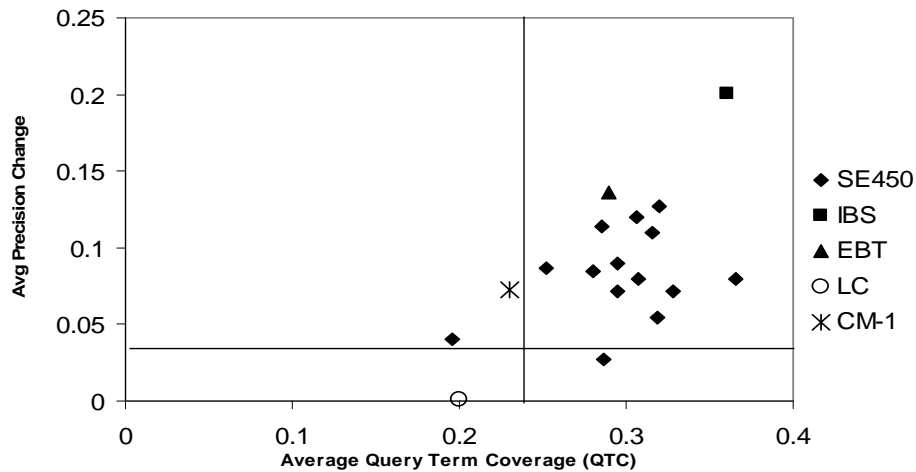
(a) QTC vs precision change by TC
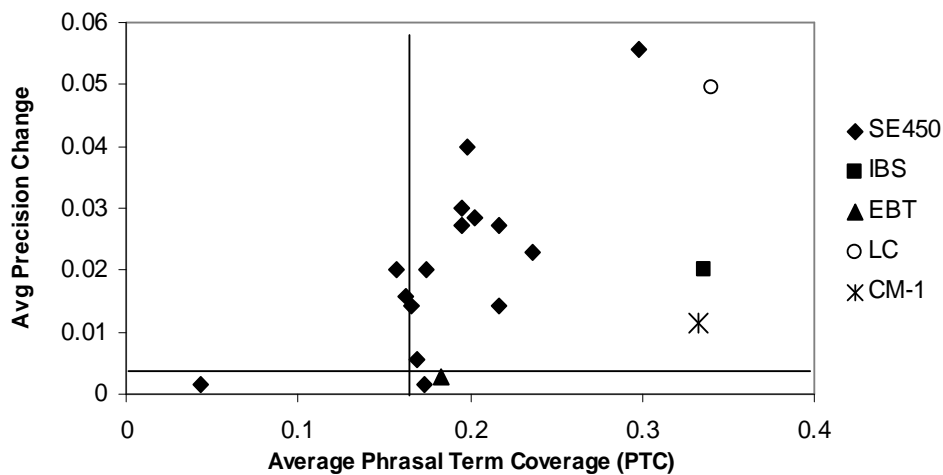
(b) PTC vs precision change by Phrasing

Figure 7.1: Association between predictors and effectiveness of enhancement methods

precision changes are typically associated to higher *Average Query Term Coverage* values. Similarly the scatterplot in Figure 7.1(b) shows a positive association between the *Average Phrasal Term Coverage* (*PTC*) metric and the average precision change achieved by the phrasing approach.

The patterns displayed in the graphs in Figure 7.1 indicate that both the TC and phrasing enhancement methods are more likely to effectively increase the precision of the retrieval results in projects that are associated with higher QTC or PTC metric values, especially for QTC values higher than 0.3 and PTC values higher than 0.2.

To illustrate how the two metrics may be used to predict the effectiveness

of the enhancement approaches, a case study was conducted on the nineteen projects. A heuristic approach was developed to define thresholds for the QTC and PTC metrics. Such thresholds can be used to determine if a given enhancement method is likely to be effective on a given project.

This approach assumes that a training set containing adequate projects with known traces between the artifacts is available. It is therefore possible to compute the precision of the retrieval results, and to identify whether a given enhancement method is effective in improving precision. The thresholding approach for QTC and PTC metrics follows a simplified version of an algorithm proposed by Cronen-Townsend et al. [13] to determine the threshold for the proposed clarity score metric measuring the performance of queries in general IR searches. In our case study, thresholds are determined using the following two-step procedure:

Step 1: Select the top 90% of projects in the training set ranked according to the average precision change achieved with the enhancement method (TC or phrasing approach). This step rejects the bottom 10% of the projects that had no significant precision change.

Step 2: Compute the predictor values (QTC or PTC) for all projects selected in Step 1 and set the metric threshold (for QTC or PTC) equal to the top 80% of the computed predictor values.

The percentiles used in the two steps were determined empirically and were tested only on the available 19 projects. Different percentile values can be selected for different projects in the training set. The approach was applied considering all 19 available projects in the training set. The horizontal lines in

Table 7.2: Leave-one-out cross-validation results using PTC values to predict Phrasing performance in the 19 available projects

Actual	Projects Predicted as		Total
	Effective	**Not-Effective**	
Effective	12	3	15
Not-Effective	2	2	4
Recall = 0.8	**Precision** = 0.86		

the graphs in Figure 7.1 (a and b) delimit the subsets created in step 1, and the vertical lines are at the thresholds computed in step 2. Based on the heuristic classification method, the TC method is expected to be effective for any project with *QTC* value of at least 0.24, while the phrasing approach is expected to improve the precision of retrieved links if a project has a *PTC* value of at least 0.17.

The heuristic approach was also validated using a leave-one-out cross validation technique that computed the threshold for the phrasing method using 18 projects in the training set and applied it to identify if the phrasing method would be effective in improving precision on the remaining one project. The process was repeated for 19 iterations and each project was used once for testing. The results are shown in Table 7.2. As there is only one available project for which the TC approach did not yield any significant improvement in precision, the classifier could not be tested on the TC approach. The heuristic procedure for computing the PTC threshold appears to be helpful, as the classifier correctly identified 80% of the projects for which the phrasing method is effective and 50% of the projects in which phrasing was not helpful. A larger size of the datasets would be needed for exploring the optimal threshold of these two metrics.

7.3 An Iterative Approach of Applying Predictors

In practice, software artifacts and associated traces are often built incrementally. This section describes an iterative technique that determines which trace retrieval approach is more effective in a new project when no prior knowledge of traces is available. The technique builds a partial answer set containing information collected from user's feedback, and then enables an enhancement strategy to be turned on or off according to feedback provided by the user. In the previous example, if phrasing had been incorrectly used on one of the projects, the iterative approach would recognize this and deactivate it as an enhancement strategy.

The iterative technique outlined in Figure 7.2 starts by utilizing the predictor values to initially select the automated retrieval algorithm for tracing in a new project. For instance if both the TC and the phrasing predictors take high values, the synergistic approach incorporating both enhancement methods will be applied to retrieve traces. The selected algorithm is then used
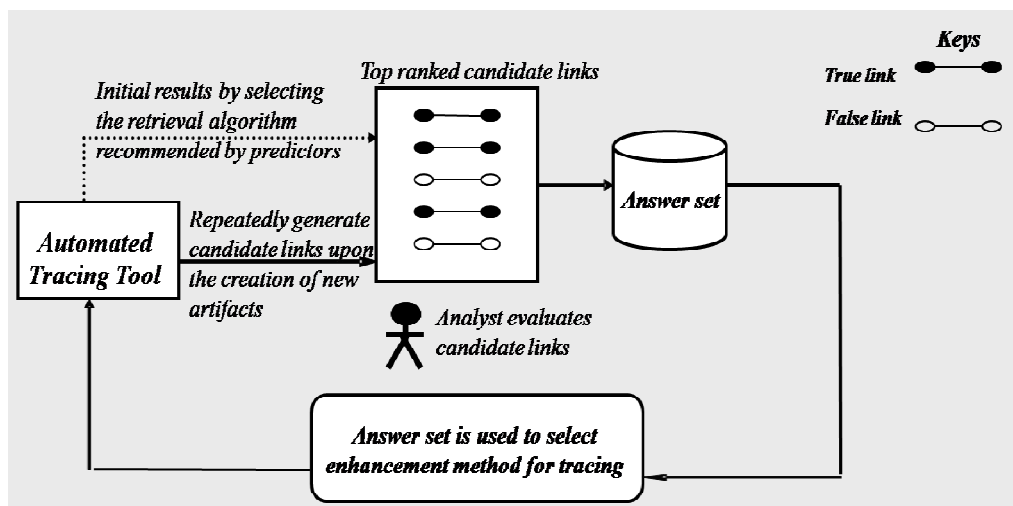
Figure 7.2: The iterative approach of applying predictors in a given project

to compute the link probability scores for the project artifacts, and the top *K* links ranked according to the probability scores are then displayed to the user for evaluation. The user identifies the true and false traces among the top K links, and the information is stored in an answer set.

In the next step the answer set is used to determine if the originally selected enhancement method is appropriate. Probability scores for each enhancement method are computed for all the known true and false traces in the answer set and compared with the basic PN model scores. An enhancement method is considered effective and will be selected for the subsequent tracing task, if the average increase in the enhancement method probability scores for true traces is larger than for false links. This metric was preferred to the standard recall/precision metrics because it is more accurate in evaluating the performance of retrieval methods for small sets of traces.

Notice that the retrieval tool is expected to be used repeatedly to generate candidate links whenever new requirements or new artifacts are created during the project development life cycle. Thus the answer set is iteratively augmented by adding user evaluations of top ranked links returned by the tool at each run.

The application of this approach is illustrated for the fifteen SE450 projects. At first 10% of the requirements were randomly selected for each project to simulate the initial state of the iterative user feedback algorithm, and then 5% of requirements were added at each later iteration. User feedback was

collected for $K=20$ or $K=50$ top ranked candidate links between the selected requirements and all traceable documents, by using the original traceability matrices supplied with the fifteen projects to simulate the user's link evaluation.

Two metrics were calculated to evaluate the accuracy of the prediction based on the answer set: *True Positive* (TP) rate that is the proportion of projects for which the enhancement methods are correctly predicted to be effective, and *False Positive* (FP) rate that is the proportion of projects for which the enhancement methods are incorrectly predicted to be effective. The results are displayed in Table 7.3. In experiments for both $K=20$ and $K=50$, when the initial answer set is small, the TP rate is about 50%, indicating that for only half of the projects is the approach able to predict correctly the effectiveness of either the TC or phrasing enhancement methods. As the answer set is augmented with additional user feedback, the FP rate decreases while the TP rate increases, indicating, as expected that a larger answer set is able to provide more accurate predictions about the enhancement methods performance. This suggests that enhancement strategies should only be activated once a sufficient body of data has been collected.

Table 7.3 Results from the iterative approach in SE450 projects

	Top 20 links								
Candidate links list size	**10%**	**15%**	**20%**	**25%**	**30%**	**35%**	**40%**	**45%**	**50%**
FP rate	0. 51	0.33	0.44	0.33	0.33	0.19	0	0	**0**
TP rate	0.51	0.82	0.74	0.85	0.74	0.92	0.89	0.93	**1**
	Top 50 links								
Candidate links list size	**10%**	**15%**	**20%**	**25%**	**30%**	**35%**	**40%**	**45%**	**50%**
FP rate	0.62	0.39	0.39	0.35	0.33	0.33	0.33	**0**	**0**
TP rate	0.58	0.90	0.93	1	1	1	1	**1**	**1**

7.4 Previously Investigated Effectiveness Factors for TC Algorithm

Two additional factors were investigated in our previous study on the effectiveness factors. These two factors, *query length* and *query word usage*, appeared to be potentially useful in predicting the effectiveness of the TC approach in a given project.

Factor #1: Query length.

The length of a query is defined as the number of non stop-words contained in it. It is intuitive that the length of a query (requirement) directly

impacts the number of its constituent terms and consequently the possible number of terms co-occurring in the documents to be traced to. Previous results of applying term coverage to TREC tasks by Rao et al. [59] also suggest that the coverage factor may not be effective on long queries. Therefore, the length of query may be a potential predictor of the effectiveness of the TC approach on individual queries.

Factor #2: Query word usage.

As described in sections 2.1 to 2.3, the standard IR models capture the similarity between a query and a traced document based on the frequency of their shared words (terms). The basic IR model would generate zero similarity score for a query and a document that have no shared terms, given that no other enhancement methods such as a thesaurus were used. Therefore the percentage of distinct terms contained in the query appearing in the whole document set is also a potential factor to investigate.

7.4.1 Impact of query length on effectiveness of TC

Query length is used as the normalization factor in the commonly used *tf-idf* term weighting schema. It is known to be an element affecting the retrieval performance of some IR models such as VSM. He and Ounis [36, 37] proposed to use query length as a predictor of query performance, i.e. whether a query would be effective in retrieving relevant documents. In their experiments they examined the linear correlation between query length and the average precision of a few TREC (Text REtrieval Conference) collections and tasks. The results however did not show significant correlation. They concluded this was due to the fact that lengths of the queries in their experiments were very similar (a variance of 0.68 terms for the short queries). The variance of the query length in our available datasets, however, is larger, with a range from 2.2 terms to 13.6 terms in different datasets. This observation indicates that the factor of query length may have more significant correlation with the retrieval precision for the requirements tracing than the TREC tasks.

In non *tf-idf* based retrieval models, query length is also believed to impact the retrieval performance. Zhai and Lafferty [78] utilized a language modeling approach to generate a language model for each document and to rank the documents by the maximum likelihood of the query based on the language

model. Smoothing methods, which are used to adjust the maximum likelihood estimator of the language model in order to improve the accuracy of the model, were compared. The results of empirical studies in which these methods were used to trace both extremely short queries (title only) and long queries (include title, description and narrative) of TREC ad hoc tasks suggested that the impact of smoothing methods on the retrieval performance is strongly correlated with query length.

An experiment is first conducted on the SE450 dataset to assess the impact of query length (factor #1) on the performance of the TC approach in a specific project. The experiment is designed to evaluate the following hypothesis:

Hypothesis #1: The TC approach achieves higher improvement on queries of shorter length than on queries of longer length.

The fourth six requirements of the SE450 dataset, whose length ranged from three to ninety two terms with a median value of six terms, are divided into two groups as shown in Table 5.4. The twenty seven requirements that contain no more than six words are labeled as short, while the nineteen other requirements are placed in the long length group. For each of the fifteen projects, the response variable is measured as the improvement in precision achieved by applying the TC approach compared to the basic PN algorithm on each group. As described in section 4.2, the TC approach produces the most improvement at the top of the candidate links list for almost all 15 SE450 projects. Therefore in this initial study the response variable is measured at recall level of 10% for the purpose of demonstration.

Table 7.4: Grouping 46 requirements of SE450 by length

	Short requirements group (length<=6 terms)	Long requirements group (length>=7 terms)
Count	27	19
Mean	4.8 terms	14.9 terms
Std Dev	1.1 terms	19.9 terms
Minimum	2 terms	7 terms
Maximum	6 terms	92 terms

The distribution of the values of the response variable for each group is depicted using box-plots in Figure 7.3. The box-plot graph visualizes the data distribution based on the five-number summary: the smallest observation, the

lower quartile, the median, the upper quartile and the biggest observation in each group. In a box-plot, the box contains the middle 50% of the data, a range between the upper quartile and the lower quartile known as the IQR (Inter-Quartile Range), and the whiskers extend to the minimum and maximum observed value. As shown in Figure 5.3, at recall level of 10%, the difference in precision for the short requirements group varies from 3% to 57% among the 15 projects, with an average of 19%. For the long requirements group, the difference in precision ranged from -15% to 26% and averaged 8%. A paired-sample t-test shows that the short requirements group has a significantly larger increase in precision than long requirements (p-value of 0.04).
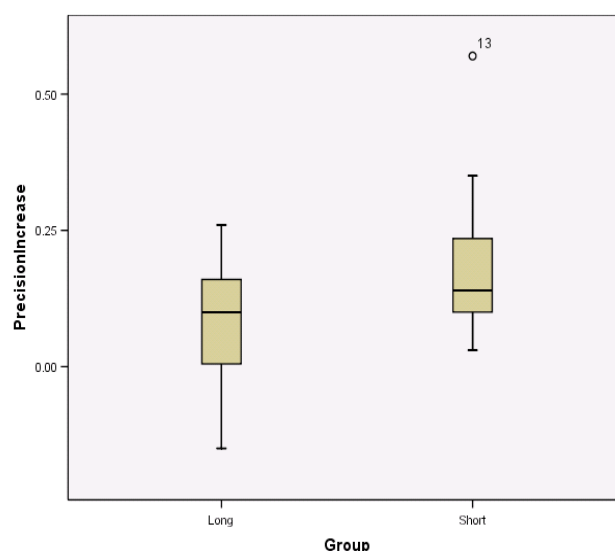
**Figure 7.3: Boxplots of Difference in Precision for
the two groups of requirements for SE450**

Note in Figure 7.3, project 13 is an outlier for the short requirement group, since the precision for that project increased by an extremely large value of 57%. The analysis of project 13 reveals a set of true links in this project containing 2-3 shared terms have been pushed significantly to the top of the candidate links list after applying the TC method. For example, the probability value between a requirement "*A road segment can originate from the intersection*" and a class "*Intersection*" is greatly enhanced because of the three shared terms "*road*", "*intersection*", and "*segment*" between them. This requirement (query) is placed into the short query groups as it contains no

more than 6 terms. This true link is assigned a relatively low probability value using the basic retrieval algorithm because the three shared terms are considered rather common in the document collection and therefore had low term weights. The change indicates that the TC approach has a strong positive effect on improving the retrieval results for this relatively short requirement. As a side note the t-test would still be significant at a P-value<0.05 if project 13 were removed from the observation.

Although the empirical study of the SE450 dataset indicates an association between query length and the performance of the TC approach, we have decided not to pursue this measure in our work on the predictors, because there is no effective way to define short and long queries.

The previous experiment on the SE450 dataset uses the median value of the query length distribution in the given dataset as the basis for the classifying criterion to define short and long queries. However, definition of short and long requirements may vary from one project to another as the traceable document types in individual projects can be different. Experiments on the query length effect in which the traceable documents are not Java source code can return different results.

There is no uniform definition of short and long queries in the general IR research either. In their study on short and long queries, Zhai and Lafferty [2001] considered queries with title only as short queries which usually contain only 2-3 terms, and queries with title, description and narrative as long queries, which may contain over one hundred terms. As the definition of short and long queries depends on characteristics of individual datasets, it is difficult to apply query length as a predictor for the effectiveness of the TC approach.

7.4.2 Impact of query word usage on TC performance

The concept *query word usage* is designed to measure whether the vocabulary of the query is consistently used in the searched document collection. More specifically, it represents the degree to which words contained in the query are used in the entire document set it searches against. Unlike *clarity score* that focuses on the performance of the IR model on the document collection with regard to a specific query, *query word usage* attempts to provide information on the potential performance of the TC

approach on a given query,

Note that this metric measures the term coverage between a query and the whole document collection. It is different from the concept of *query term coverage* introduced earlier in chapter 4, which is a measurement between individual query-document pairs.

Query word usage definition

Let q be a query consisting of k distinct single terms/words $\{t_1, t_2,..., t_k\}$ that trace to a set of documents $D=\{d_1, d_2, ..., d_m\}$. *Query word usage* between q and D, *WU(q,D)* can be defined as follows:

$$\mathrm{WU(q,D)} = \frac{\sum_{i=1}^{k} I_A(t_i)}{k}$$

where $I_A(t_i)$ is the indicator function defined over a set A which includes the vocabulary of the whole document collection. $I_A(t_i)$ is defined as $I_A(t_i) = \begin{cases} 1, & t \in A \\ 0, & t \notin A \end{cases}$. In other words, $I_A(t_i)$ takes value 1 when the query term t_i is also used in one or more of the documents and value 0 when the term does not appear in any document.

Query word usage reaches the maximum value of 1 when all distinct terms in the query are found in the document set.

Effect of query word usage on TC performance

To examine the effect of query word usage on the performance of the TC approach, an experiment is conducted on the SE450 dataset in which the traced documents are Java classes. The dataset is selected because all the 15 projects in this dataset implement the same set of requirements so the *query word usage* of these requirements is only affected by the different Java code documents in each student project, and experimental error due to the variance within the requirements is eliminated.

The experiment was designed to examine the following hypothesis:

Hypothesis #2: The TC approach achieves higher improvement on queries with higher query word usage than on queries with lower query word usage.

The *query word usage* of the forty six requirements in the Java classes is calculated with respect to each of the fifteen SE450 student projects. In each project, the forty six requirements are then divided into three groups, labeled
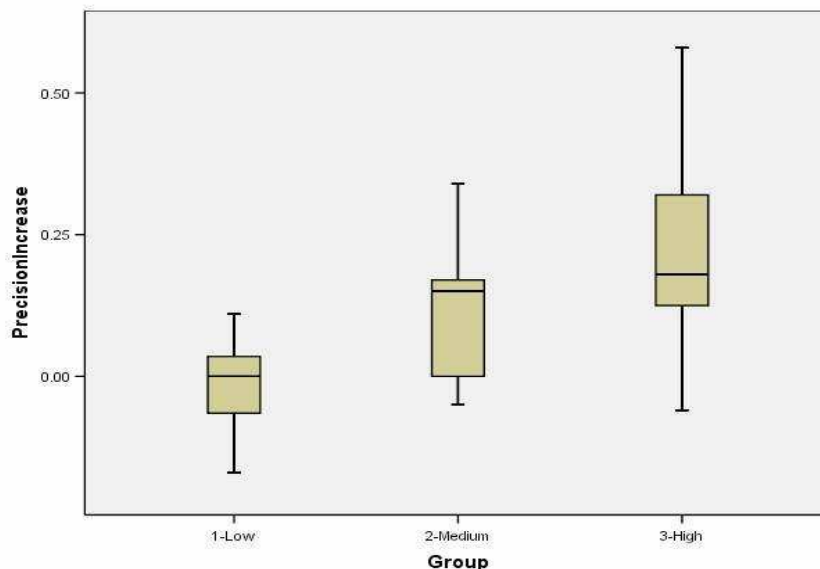
**Figure 7.4: Boxplots of difference in precision at top
for three groups of requirements**

as *low*, *medium* and *high* based on their *query word usage* in the Java classes
of that project. Requirements of no more than 1/3 word usage are placed in the
low usage group. For the remaining requirements, those whose query word
usage is less than 1/2 are placed into the medium usage group while all others
are in the high usage group.

Similarly to the experiment on the factor of query length, the response
variable is measured as the difference in precision after applying the TC
approach compared to the basic IR algorithm at the recall level of 10% for
each of the three groups.

As displayed in Figure 7.4, the difference in precision after applying the
TC approaches on requirements with low query word usage ranges from -17%
to 15%, with an average of -0.7%. For requirements with medium query word
usage, the difference in precision is generally higher, ranging from -5% to
34% with an average of 11%. Precision change varies from -6% to 58% on
requirements with high query word usage and averages 22%.

A one-way ANOVA is performed on the three groups and the results reveal
that the three groups have significantly different average change in precision
(p-value<0.0001) after using the TC approach. A pair-wise comparison using
Fisher's Least Significant Different test (LSD) is computed between each pair
of groups. The results indicate that the means in any two groups are

significantly different from each other. Requirements with high query word usage achieve significantly higher average precision change than requirements with medium query word usage, and requirements with medium query word usage achieve significantly higher average change in precision than requirements with low query word usage.

Table 7.5: SPSS Linear Regression Output
Model Summary

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate	Change Statisitcs				
					R Square Change	F Change	df1	df2	Sig. F Change
1	.599 (a)	.359	.344	.12834	.359	24.107	1	43	.000
a Predictors: (Constant), AvgWordUsage									

Coefficient (a)

Model		Unstandardized coefficients		Standardized coefficients	t	Sig
		B	Std. Error	Beta		
1	(constant)	-.122	.050		-2.410	.020
	AvgWordUsage	.541	.110	.559	4.910	.000
a Dependent Variable: Precision Increase						
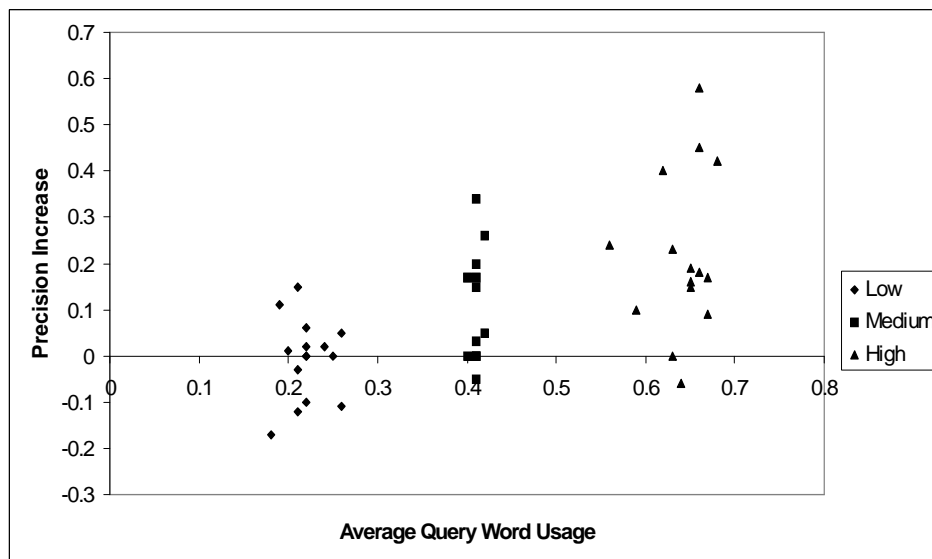
Figure 7.5: Association between average query word usage

The results of the statistical analysis suggest that the *query word usage* appears to have significant effect on the precision change caused by applying the TC approach. As a precursor of investigating *query word usage* as a predictor of the strength power of the TC approach, an exploratory analysis is

conducted on the SE450 dataset to explore the correlation between *query word usage* and precision improvement. In each of the 15 projects, the average *query word usage* of the three groups, low, medium and high usage are obtained individually and depicted on the x-axis, while their according average precision change after applying the TC approach are depicted on y-axis, as shown in figure 7.5. There seems to be a relatively strong linear correlation between average query word usage and precision change at the top.

Assuming there is a linear relationship existing between the two variables, we make an attempt to fit the data points into a linear regression model where query word usage is the independent variable (predictor) and precision improvement is the response variable. From SPSS output (displayed in Table 7.5), the coefficient β_1 for the predictor in this model is estimated as 0.541 which showed a moderately positive correlation between precision change and query word usage.

However when the response variable is measured as the average precision change at different recall levels after applying the TC approach compared to the basic IR algorithm, which is considered a more accurate metric for the effectiveness of the enhancement method, query word usage appears to be a weak predictor for the TC approach. As displayed in the boxplots in Figure 7.6, the difference in the average precision change after applying the TC approaches on requirements with low query word usage ranges from -3% to 20%, with an average of 3%. For requirements with medium query word usage, the average precision difference ranges from -5% to 15% with an average of 6%. Precision change varies from -6% to 15% on requirements with high query word usage and averaged 4%.

A one-way ANOVA is then performed on the three groups and the results reveal that the null hypothesis that there are no significant difference in the average precision change among the three groups after using the TC approach can not be rejected (p-value=0.18).
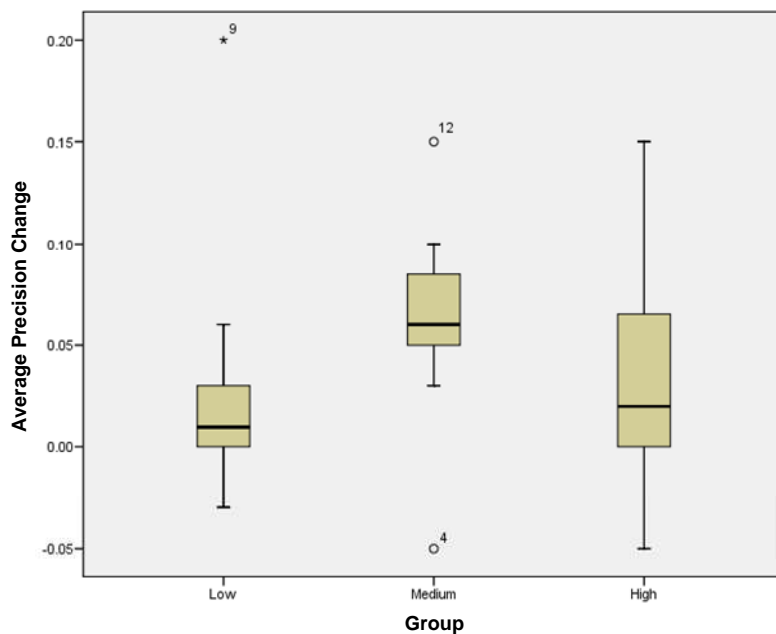
Figure 7.6: Boxplots of difference in average precision change
for three groups of requirements by using TC

Hence we conclude that the association between query word usage and precision improvement by applying the TC approach is not significant enough. We therefore decide not to pursue *query word usage* as a potential predictor for the effectiveness of the TC approach.

CHAPTER 8: EVALUATING EFFECTIVENESS OF PROJECT GLOSSARIES FOR TRACE RETRIEVAL

The experiments discussed in section 6.4 of chapter 6 show the effectiveness of the project glossary approach in improving precision for the top ranked retrieved links as well as its limitations and constraints. Such an approach can only be successfully applied to projects for which a project glossary exists that was used consistently throughout the design of the software artifacts. In reality many software projects do not have a pre-constructed glossary available or have a "weak" glossary that has not been consistently used. Therefore it would be helpful to predict the effect of the project glossary usage in link retrieval prior to running the automated trace retrieval technique. The following research problems are investigated in this chapter:

1. What characteristics must an existing project glossary have in order to be potentially useful for improving the retrieval results?

2. For projects in which a glossary is unavailable, can a set of key terms and phrases be extracted and used in lieu of the project glossary to help improve the retrieval results?

To answer the first question, an empirical study is conducted to analyze the retrieval results using existing project glossaries. The study identifies certain project glossary characteristics that could be used to predict when the glossary approach can be effective in improving the tracing results precision for a specific project.

This chapter describes the empirical study in section 8.1 and introduces a set of criteria for evaluating the usefulness of an existing project glossary in the glossary approach. Section 8.1.1 evaluates the criteria against the IBS and SE450 datasets for which the project glossary is available. As part of this study, a procedure is also presented in section 8.2 to automatically extract critical keywords and phrases from the requirements collection of a given project. The extracted items are then used to enhance the automated trace retrieval algorithm in lieu of a missing or unavailable project glossary. The experiments are described in section 8.2.1.

8.1 Criteria to Evaluate Project Glossaries for Trace Retrieval

Criterion # 1: Project glossary items should be consistently used in the traced documents.

A project glossary describes the terminology used in a specific software project, and is created to facilitate a consistent use of terms in the project artifacts during the development phases. However in practice, project glossaries are often not consistently followed and synonyms of glossary terms are used instead in requirements specifications or software artifacts. When this happens, project glossaries may have insignificant or no impact on the retrieval results.

The presence of synonyms of glossary items in the traced documents provides a strong indication that the glossary may not have been consistently followed in the project development. A simple method to detect synonyms of glossary terms is implemented using WordNet [82]. WordNet is a semantic dictionary in which words are organized into logical groups called '*synsets*' that consist of related synonyms. Besides synonyms contained in the same synset, a pair of words in which one word is the direct hyponym (a word whose semantic range is within that of another word) are also considered synonyms in our experiment. For example, "*car*" and "*vehicle*" will be considered synonyms as "*car*" is the direct hyponym of "*vehicle*".

Criterion # 2: Glossary items should have high term specificity.

Term specificity indicates the quality of a term in describing the document content, and is commonly computed using *idf* (*inverse document frequency* [83]. Thus glossary terms specificity is measured as $idf(t)=ln(|R|/|R_t|)$, where $|R|$ is the total number of requirements and $|R_t|$ is the number of requirements containing t. Glossary terms with high specificity values occur in fewer requirements, and are more useful to identify a specific concept, and hence to retrieve documents related to that concept.

Criterion # 3: Glossary items should be domain specific.

Domain-specific terms occur more frequently in project specific documents, and are often associated with critical concepts of the project. The domain specificity $DS(t)$ for a term t is computed as follows:

$$DS(t)=\ln\left(\frac{freq(t,R)}{\sum_{t\in D} freq(t,R)} \middle/ \frac{freq(t,G)}{\sum_{t\in G} freq(t,G)}\right) \qquad (8.1)$$

where *freq(t,R)* is the frequency of term *t* in the requirements collection *R* associated with the project glossary, and *freq(t,G)* is the frequency of term *t* in the general technical corpus *G* that contains requirements from various domains. If a term is unique to the specific project, i.e. *freq(t,G)=0, DS(t)* is assigned a large value.

In our experiments the general corpus G contains the requirements from all other available software projects. In our experiments the corpus contains thirty eight sets of Software Requirement Specifications (SRS) taken from a variety of software projects. In addition to the five datasets that are introduced in section 3.1, the corpus also includes projects ranging from industrial applications to research projects. Project topics include NASA's Moderate Resolution Imaging Spectrometer, an industrial production lines construction system, vehicle parts finder, meetings scheduler, battleships game, and an enterprise level service bus scheduling system.

8.1.1 Applying the criteria to project glossaries

The three criteria are applied to evaluate the impact of a project glossary in increasing the precision of the retrieval results. Thus the project glossary can be considered "weak" with respect to its ability to improve the retrieval results for the project if synonyms of glossary terms are used, and/or glossary terms have average low specificity and low domain-specificity.

The IBS project and the fifteen SE450 projects were the only projects supplied with a project glossary. The IBS glossary has six keywords and twenty eight phrases, while the SE450 glossary contains four keywords and six phrases. On the basis of the proposed criteria, the SE450 project glossary was found to be inconsistently used in the fifteen projects, while the IBS project glossary was found to be more meaningful. Synonyms of glossary terms were frequently used in the SE450 Java classes, for instance '*car*' and '*obstruction*' were used in place of '*vehicle*' and '*obstacle*'. No synonyms of glossary items were detected in the IBS dataset. Both the average term

specificity and the average domain specificity of the SE450 glossary items were significantly lower than the corresponding values for IBS glossary terms [80].

The weakness of the SE450 projects glossary is confirmed by the results in section 6.4 for the SE450 datasets which show that the glossary approach reduced the accuracy of the tracing tool retrieval results. On the contrary for the IBS dataset, the glossary approach was able to retrieve a larger proportion of correct traces. These results suggest that the three proposed criteria provide a simple effective way to predict when a project glossary can be effectively used to improve the accuracy of the retrieval tool.

8.2 A Method for Automatically Extracting Keywords and Phrases

This section presents an automated technique for extracting a set of important keywords and phrases from the project requirement specifications that can be used in lieu of project glossaries to help improve the precision of the retrieval algorithm. This technique can be used when glossaries are either not supplied with the software projects or are inconsistently followed during the development phase.

Several methods for extracting keywords from documents collections have been proposed in IR. Some techniques are based on statistical approaches that identify significant terms on the basis of term frequency [84], but ignore the syntactical meaning of the terms. Our proposed extraction method applies a syntactical method to identify critical keywords and phrases from the requirement specifications. The syntactical approach enables the tool to extract only single nouns and two-noun phrases that are most common in a project glossary. The approach consists of the following two steps:

Step 1: Generation of candidate keywords and phrases. Candidate items including single nouns and two-noun phrases are identified by using a POS tagger such as Qtag on the set of requirements specifications.

Step 2: Filtering. Filters are applied to remove unimportant items from the list generated in Step 1. The following three filters are applied:

Filters A and B: Term and Domain specificity. Keywords with Term and Domain specificity values below certain thresholds will be removed, as they might decrease the precision of the trace retrieval

results. The threshold values are set by the analyst and depend on document collection characteristics.

Filter C: Nouns filtering. A single noun that is included in a candidate phrase as head noun is removed as the phrase is considered more meaningful to represent specific concepts. For example, '*truck*' in the phrase "*truck list*" is used to modify the head noun "*list*". It is necessary to remove "*list*" from the candidate terms set as "*truck list*" is considered more specific than single noun "*list*",

The resulting list will consist of single nouns and two-noun phrases that have high term and domain specificity.

8.2.1 Evaluating the extraction method

The automated extraction method is applied to the requirements collections for the SE450 dataset and the EBT, LC and CM1 datasets. In our experiments, Filter (A) in step 2 removes a low term specificity keyword if it occurs in at least three requirements for projects containing less than 60 requirements, or in at least 5% of the total requirements for larger projects. The filtering step (B) based on domain specificity retrieves 1) all unique items for the project (i.e. *freq(t,G)*=0), and 2) the top 50% of the items with the highest domain-specificity score. The thresholds in filtering A and B are selected on the basis of an exploratory study that analyzes the distribution of glossary terms in the requirements documents for the available projects. Most of the items in the IBS project glossary occur in no more than three requirements, while many items of the 'weak' project glossaries associated with the SE450 dataset occur in a much larger number of requirements. Exploratory results suggest that the selected threshold values are appropriate for extracting critical phrases and keywords to be effectively used to improve the accuracy of the retrieval approaches.

The extracted set for the SE450 dataset contained five keywords and 30 phrases, with only two terms in common with the existing project glossary. For EBT and LC, the set contains eight keywords and 15 phrases, and six keywords and 11 phrases respectively. For the large-scale CM1 dataset, the method extracts 59 keywords and 164 phrases.

The project glossary approach is then applied to the datasets using the extracted critical set. The two graphs in Figure 8.1 display changes in

(a) compared to basic PN

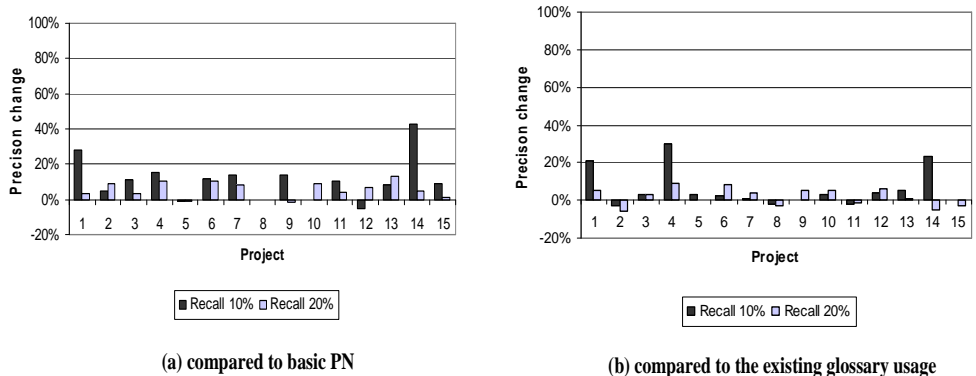(b) compared to the existing glossary usage

**Figure 8.1: Precision change at 10%, 20% recall
for algorithm using extracted set on SE450**
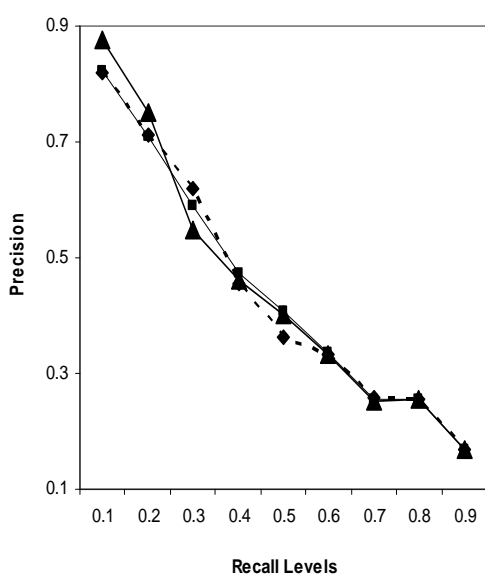
Figure 8.2: Effect of extracted set on EBT
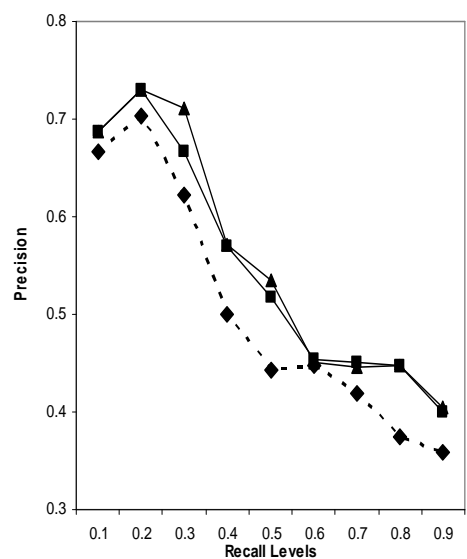
Figure 8.3: Effect of extracted set on LC
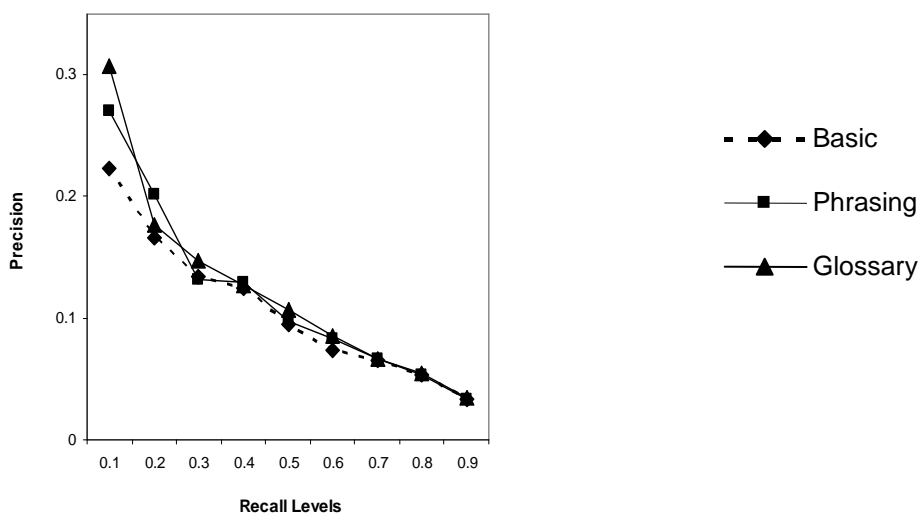
Figure 8.4. Effect of extracted set on CM-1

precision at 10% and 20% recall for the fifteen SE450 projects comparing the Project Glossary approach incorporating the extracted keywords set with (a) the basic PN and the Phrasing algorithms, and with (b) the Project Glossary algorithm using the existing project glossary. Both graphs show that the extracted keywords set is more effective in improving the results precision, and, when compared with the existing project glossary, provides more meaningful information for identifying correct traces. Similarly, results for the EBT, LC and CM1 datasets displayed in Figures 8.2, 8.3 and 8.4 show that the extracted keywords set is effective in improving the retrieval accuracy, especially among the top retrieved links.

The effect of applying the Project Glossary approach using the extracted set in the SE450 dataset suggests that the extracted keywords and phrases maybe more meaningful than the terms defined in the existing glossary and can achieve more accurate retrieval results. The approach is then also evaluated against the IBS dataset for which the existing project glossary has proven to be useful in improving the retrieval results, as described in Chapter 6.

Compared to the existing glossary which contains 34 entries (six keywords and 28 phrases), the extraction method identifies a larger set of 117 entries (32 keywords and 85 phrases) from the requirement specification of the IBS dataset. The majority of the items contained in the extracted set are not defined in the glossary. More specifically, the extraction method detects 30 new keywords and 69 new phrases.

Figure 8.5 illustrates the effect of applying the glossary approach using the extracted critical set in the IBS compared to the other retrieval algorithms. As clearly shown in this figure, the synergistic algorithm that incorporates TC, PH and the glossary approach utilizing the extract set yields the highest retrieval accuracy at most of the recall levels. Compared to the synergistic algorithm that utilizes the existing project glossary, this new algorithm yields additional 3% of increase in the precision at 10% recall level and 4% at 20% recall level. The observation of the extracted key set in the IBS reveals that the set eliminates some keywords and phrases that are defined in the project glossary, as they are identified as terms with low term specificity or low domain specificity. Examples include glossary items "schedule" and "truck" which

occur in multiple requirements and therefore are associated with relatively low term specificity. The probability scores of a few false links containing these
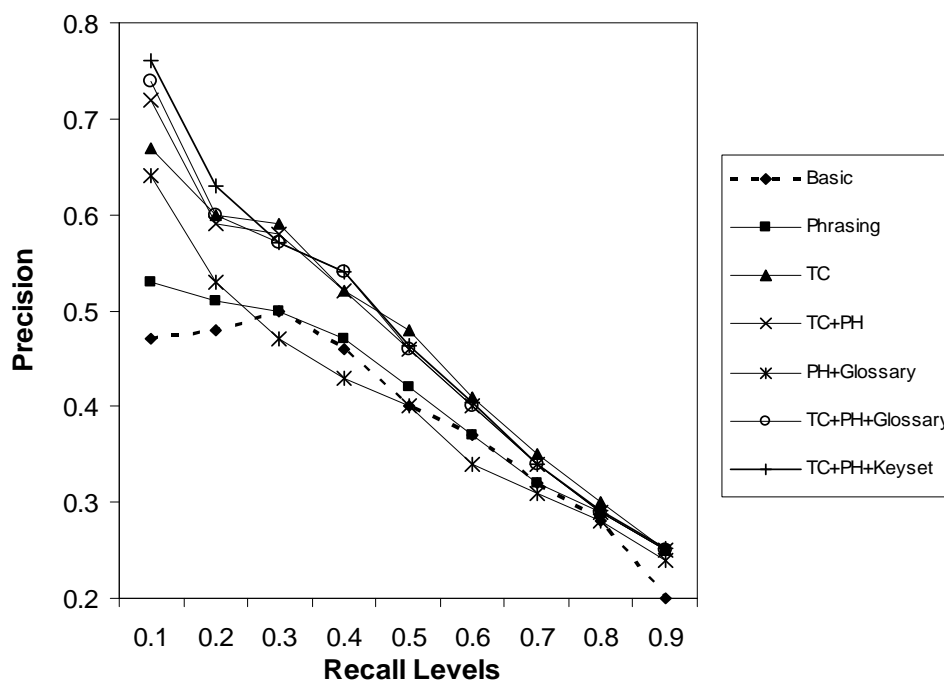
Figure 8.5: Effect of applying extracted key set in IBS

"weak" terms would have been incorrectly enhanced after applying the glossary approach. By applying the glossary approach incorporating the extract set, theses false links will not be enhanced. Therefore more false links can be eliminated from the candidate links list by setting a higher threshold for the tracing tool, and consequently the retrieval accuracy can be improved.

The results in the SE450 and IBS datasets indicate that the automated extraction method can effectively identify a set of keywords and phrases that are critical to a project. The generated keywords and phrases set could be more useful than the existing project glossary in improving the trace retrieval results.

CHAPTER 9: CONCLUSIONS

9.1 Discussion on Validity

This section discusses three factors that may affect the generalization of our results and conclusions presented in this thesis: the accuracy of the retrieval results, the datasets used in our experiments and the use of the enhancement predictors.

1. Accuracy of the trace retrieval results

The accuracy of the tracing results in a specific project is evaluated using a pre-defined 'answer set' that defines the set of traceability links between the artifacts in this project. The validity of our conclusions and the results of our experiments therefore are affected by the correctness of the answer sets.

As described in Chapter 3, the answer set for each of the five datasets was originally built by software engineers who manually traced the artifacts in the dataset. For example, the answer set for the CM-1 dataset was created by NASA software engineers who had deep understanding of the software system. There is always a certain level of subjectivity in the evaluation of traces, and it is possible that analysts may have missed some traces, or have incorrectly included irrelevant traces. To mitigate the risk of the omission of true traces and the inclusion of false traces in the answer sets, the trace matrices of all five datasets were refined and re-evaluated thoroughly by several researchers before conducting the traceability experiments. However, it is still possible that the answer sets are either incomplete or contain incorrect traces. Thus the recall and precision values in our experiments should be more precisely interpreted in terms of the traces identified as "correct" through manual tracing.

2. Datasets used in experiments

Another factor that might affect the validity of our results is the limited number of available datasets against which our approaches are evaluated. The effort involved in manually building the answer set for a specific project is significant and the process is lengthy. As an example, it took nearly two months for two researchers in our group to validate and correct the answer sets for the fifteen projects in the SE450. As a result, it is difficult for us to collect and create more datasets for the studies.

Currently in the five datasets available to us, the traceability experiments are focusing on traces between high-level and low-level requirements, between requirements and UML design elements, and between requirements and source code. At this point we are unable to evaluate our approaches to other document types since we do not have additional datasets available to us. It would be interesting to investigate how the effectiveness of the three enhancement methods and the enhancement predictors may vary over different document types.

3. Use of enhancement predictors

The heuristic approaches of using the two predictors QTC and PTC that are described in Chapter 7 might also affect the validity of our results.

The first approach that identifies the threshold for the predictor values assumes that projects have known traces. This may have limited practical value, since traces are often not available. The leave-one-out cross-validation procedure is an attempt to check the validity of the threshold value for the PTC metric. The validation on the QTC was not conducted as there is only one project among the fifteen for which the TC approach is not effective. The results of this validation procedure might show some bias since fifteen of the nineteen projects have strong commonalities. They are fifteen student projects produced as part of the same course, and with the same set of requirements. Additionally, the threshold values identified in our experiments have not been tested on other projects. This may increase the risk of the prediction models overfitting the five datasets available to us.

Another concern relates to results for the iterative procedure to select the effective enhancement procedures. User feedback is simulated using the known answer set, and therefore it does not consider that the analyst may miss some links, or trace incorrect artifacts during the manual tracing process. The results from the iterative approach described in Table 7.3 could therefore overestimate the accuracy of the iterative procedure at the various steps, and less accurate feedback could result in more frequent switches between enhancement strategies.

9.2 Summary of Contribution

The primary goal of the work presented in this thesis is to improve the

precision of the automated requirements trace retrieval results by utilizing IR techniques, and at the same time to maintain a high recall in the results. As discussed in Chapter 1, the low precision problem associated with the IR-based automated tracing tools, regardless of the IR models implemented in the tools, would force the analyst to manually filter out a large number of unwanted links in the trace retrieval results. The low precision in the retrieval results has become a problem and would apparently decrease the trust of the analyst in the usefulness of the automated tracing tools, and consequently impact the adoptability of the IR-based tracing tools in industry.

In order to achieve the goal of improving the precision of the trace retrieval results, this thesis has made the following major contributions:

1. **Presented and evaluated three enhancement strategies that can be incorporated either individually or synergistically into the basic IR model.**

Compared to the methods previously proposed by other researchers to improve the retrieval results [9, 33, 35], these three methods, TC, Phrasing, and Project Glossary, are easier to implement and require no extra human effort. The validation of the three enhancement strategies is focused on a PN (Probabilistic Network) model. However these methods can also be easily applied to other IR models such as VSM and LSI as a way to improve the standard *tf-idf* term weighting strategy.

The results reported in the thesis have been mixed, as the performance of these retrieval algorithms varies from project to project; however they have proven the general effectiveness of using such enhancement methods to improve the retrieval results especially among the top retrieved links. As discussed earlier in Chapter 3, the top retrieved links contain the links that will be seen and inspected by the analyst first. Therefore the improved precision at the low recall levels is especially meaningful as it implies more true links are listed at the top of the retrieval results and the analyst may be able to find those important links earlier in the process.

Among all methods, the synergistic application of TC and phrasing together is shown to be the most effective and significantly improves precision of the top retrieved links for almost all datasets. When a meaningful project

glossary is available, the synergistic application of the TC, phrasing and project glossary methods appear to yield the highest increase in precision among the top ranked retrieved links. The impact of the synergistic approaches that incorporate multiple enhancement methods seems to be closely related to the effectiveness of individual enhancement methods. For instance, when the project glossary approach has a negative effect on the precision of the retrieval results, the synergistic approach may also decrease the precision of the results.

The TC method (algorithm T) shows consistent improvement in precision when applied across all datasets compared to the basic PN model. When applied with phrasing, this approach was able to increase the precision among the top retrieved links significantly in some of the datasets.

The phrasing algorithm achieved considerable improvement on precision for most of the datasets. The improvement resulting from phrasing is generally less significant than the TC method. Some projects even experienced a decrease in precision when phrasing was applied. Similarly the effect of applying the glossary approach has been mixed. The analysis of these traces motivated the work of investigating characteristics of software projects that can explain differences in performance of the enhancement strategies.

Once constraint associated with these term-based strategies is that they are less effective in increasing the precision at high recall levels, since the low-ranked missed traces are often between requirements and documents that share very few or no terms or phrases, and therefore term-based approaches are not able to retrieve them.

2. **Investigated and evaluated a procedure to automatically extract critical keywords and phrases from the requirements collection of a given project; the extracted items are then used to enhance the automated trace retrieval algorithm.**

The results of applying the glossary approach reported in this thesis have not been consistent. The analysis of the traces retrieved for some projects in which the project glossary approach was not effective, revealed that the decrease in precision was due to some glossary items being used inconsistently in the documents collection.

An extension to the project glossary retrieval algorithm is then explored to

overcome some limitations of the project glossary approach. The thesis presents some techniques to evaluate the quality of information in a project glossary and the usefulness of these techniques in improving trace retrieval results. Occurrence of synonyms in the traceable documents and frequency of project glossary terms may provide information on the meaningfulness of a project glossary to describe critical concepts. More specifically, the project glossary can be considered "weak" with respect to its ability to improve the retrieval results for the project if synonyms of glossary terms are used, and/or glossary terms have low average specificity and low domain-specificity.

The developed criteria were validated against a few dataset. These results suggest that the three proposed criteria provide a simple effective way to predict when a project glossary can be effectively used to improve the accuracy of the retrieval tool.

Furthermore, the thesis has also discussed an automated method to extract keywords and phrases from the existing requirements collection in a project with 'weak' glossary or no available glossary. The method was evaluated against two datasets containing a project glossary, and the results show that the extracted keywords set is more effective in improving the precision of the results. When compared with the existing project glossary, the extracted keywords set is found to provide more meaningful information for identifying correct traces.

The retrieval algorithm using either the project glossary information or the set of extracted key terms and phrases achieves high precision among the top ranked retrieved links. The enhancement methods presented in this thesis can increase the analyst's trust in the tracing tool. The constraint of this approach is that some true links might still be missed, and are hard to be retrieved using only textual content information.

3. **Presented and evaluated prediction models using the predictors for the enhancement methods that can be applied in automated tracing tools for better retrieval results.**

Two predictors, *average Query Term Coverage* (*QTC)* and *average Phrasal Term Coverage* (*PTC)*, are explored as the effectiveness measure for the TC and the phrasing methods respectively. The experimental results

indicate that both the TC and phrasing enhancement methods are more likely to effectively increase the precision of the retrieval results in projects that are associated to higher QTC or PTC metric values, especially for QTC values higher than 0.3 and PTC values higher than 0.2.

With the assistance of the prediction models that utilize the presented predictors for individual enhancement methods, an automated tracing tool can make real-time decisions on whether to apply a certain method in order to achieve the best retrieval results. Results of a small-scale study indicate that the predictor values can provide useful guidelines for selecting a specific tracing approach when there is no prior knowledge of the 'answer set' for a given project.

The performance of the prediction models can be improved by learning from user feedback. Although the approach may require additional human effort, it is practical considering that the user is to interact with the tracing tool and build the implicit trace matrix over time. Furthermore, the benefits of improving the precision in a given project will be experienced throughout the remaining lifetime of the software system, and could significantly alleviate future maintenance efforts.

To summarize, the work presented in this thesis supports the development and application of automated tracing tools. The three strategies, *TC*, *Phrasing*, and *project glossary* share the same goal of improving precision in the retrieval results to address the low precision problem, which is a big concern associated with the IR-based tracing methods. Furthermore, the predictors for individual enhancement strategies presented in this thesis can be utilized to identify which strategy will be effective in the specific tracing tasks. These predictors can be adopted to define intelligent tracing tools that can automatically determine which enhancement strategy should be applied in order to achieve the best retrieval results on the basis of the metrics values. A tracing tool incorporating one or more of these methods is expected to achieve higher precision in the trace retrieval results than the basic IR model. Such improvement will not only reduce the extra effort required for the analyst to inspect the retrieval results, but also increase his or her confidence in the accuracy of the tracing tools and consequently help build the analyst's trust in

the tools.

The term-based enhancement approaches have their own limitations. They are typically not as effective in increasing the precision at the bottom of the retrieval results, since requirement-document pairs assigned with low similarity scores share very few or no terms or phrases, and therefore are hard to retrieve using only term-based approaches. Alternative IR approaches that use additional information, such as hierarchical structure of the software artifacts [9] to discover relations between documents should be investigated. Such approaches can be used jointly with the term-based enhancement strategies to improve the overall accuracy of the retrieval results, especially for the traces that are typically missed by the text-based retrieval approaches.

REFERENCE

[1] Antoniol G., Canfora G., De Lucia A, Casazza G, "Information Retrieval Models for Recovering Traceability Links between Code and Documentation", *Proceedings of the International Conference on Software Maintenance*, San Jose, California, USA, 2000, pp. 40-51.

[2] Antoniol G., Canfora G., Casazza G., De Lucia A, Merlo E., "Recovering Traceability Links between Code and Documentation", *IEEE Transactions on Software Engineering*, 28(10), 2002, pp. 970-983.

[3] Borger E, Gotzhein R, "Requirements Engineering Case Study 'Light Control'", *Journal of Universal Computer Science,* 6(7), 2000, pp. 580-596.

[4] Buckley C., Singhal A., Mitra M., Salton G., "New retrieval approaches using smart: TREC-4", *Proceedings of the Fourth Text REtrieval Conference* (TREC-4), 1995, pp. 25-32.

[5] Burke R., Hammond K., Kulykin V., Lytinen S., Tomuro N, Schoenberg S, "Natural language processing in the FAQ finder system: results and prospects". *Working Notes of the Spring AAAI Symposium on the WWW*, Menlo Park, CA, 1997, pp. 26-33.

[6] Church K, Hanks P, "Word Association Norms, Mutual Information, and Lexicography", *Computational Linguistics*, 16(1), 1990, pp. 22-29.

[7] Cleland-Huang J, Chang C.K, Sethi G, Javvaji K, Hu H, Xia J. "Automating Speculative Queries through Event-based Requirements Traceability*", Proceedings of the IEEE Joint International Requirements Engineering Conference*, Essen, Germany, 2002, pp. 289-298.

[8] Cleland-Huang J, Settimi R, BenKhadra O, Berezhanskaya E., Christina S, "Goal-Centric traceability for managing non-functional requirements", *Proceedings of the 27th International Conference on Software Engineering*, St. Louis, MO, 2005, pp. 362-371.

[9] Cleland-Huang J., Settimi R., Duan C., Zou X., "Utilizing supporting evidence to improve dynamic requirements traceability", *Proceedings of the 13th IEEE International Requirements Engineering Conference*, Paris, France, 2005, pp. 135-144.

[10] Cleverdon C. W, Keen E. M, *Factors Determining the Performance of Indexing Systems*, Cranfield, England: College of Aeronautics, 1966.

[11] Cover T. M, Thomas J. A, *Elements of Information Theory*, New York: Wiley-Interscience, 1991.

[12] Croft W., Turtle H., Lewis A, "The use of phrases and structured queries in information retrieval*", Proceeding of the 14th International ACM SIGIR conference on Research and development in information retrieval*, Chicago, IL, 1991, pp. 32-45.

[13] Cronen-Townsend S., Zhou Y., Croft W. B, "Predicting query performance*", Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, Tampere, Finland, 2002, pp 299-306.

[14] Davis A.M, *Software Requirements: Analysis and Specification*, Englewood Cliffs, NJ: Prentice Hall, 1990.

[15] Davis, A.M, "The analysis and specification of systems and software requirements", *Systems and Software Requirements Engineering*, IEEE Computer Society Press, 1990*, pp.119–144.

[16] Dillon M., Gray A.S, "FASIT: A Fully Automatic Syntactically Based Indexing System", *Journal of the American Society for Information Science*, 34, 1983, pp. 99-108.

[17] Domges R., Pohl K, "Adapting Traceability Environments to Project Specific Needs", *Communications of the ACM*, 41(12), 1998, pp. 55-62.

[18] Egyed, A, "A Scenario-Driven Approach to Traceability", *Proceedings of the 23rd International Conference on Software Engineering (ICSE 2001)*, Canada, 2001, pp 123-132.

[19] Egyed A., Gruenbacher P., "Automatic Requirements Traceability: Beyond the Record and Replay paradigm*", Proceedings of the 17th IEEE International Conference on Automated Software Engineering (ASE),* Edinburgh, UK, 2002, pp. 163-171.

[20] Egyed A., "A Scenario-Driven Approach to Trace Dependency Analysis", *IEEE Transactions on Software Engineering*, 9(2), February 2003, pp. 116-132.

[21] Evans D. A., Zhai C., "Noun-phrase analysis in unrestricted text for information retrieval*", Proceedings of the 34th Annual Meeting of the*

Association for Computational Linguistics, Santa Cruz, CA, 1996, pp. 17-24.

[22] Evans M.W, *The Software Factory*, Hoboken, NJ: John Wiley and Sons, 1989.

[23] Fagan J, "Experiments in Automatic Phrase Indexing for Document Retrieval: A Comparison of Syntactic and Non-Syntactic Methods", (Doctoral dissertation, Cornell University, Computer Science Department, 1987), *Technical Report*, 1987, pp. 87-868.

[24] Forsythe G.E., Malcolm M.A., Moler C.B, *Computer Methods for Mathematical Computations* (Chapter 9: Least squares and the singular value decomposition), Englewood Cliffs, NJ: Prentice Hall, 1977.

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

[26] Gay L., Croft W, "Interpreting Nominal Compounds for Information Retrieval", *Information Processing and Management,* 26(1), 1990, pp. 21–38.

[27] Gotel O., Finkelstein A, "An analysis of the requirements traceability problem*", Technical Report TR-93-41, Department of Computing*, Imperial College of Science Technology and Medicine, London, UK, 1993.

[28] Gotel O, Finkelstein A, "An analysis of the requirements traceability problem", *Proceedings of the 1st International Conference on Requirements Engineering*, Colorado Springs, Colorado, 1994, pp. 96-101.

[29] Gotel O, Finkelstein A, "Contribution Structures", *Proceedings of the 2nd IEEE International Symposium on Requirements Engineering*, York, England, 1995, pp. 100-107.

[30] Grishman R, *Information extraction: Techniques and challenges* (*Lecture Notes in Computer Science*), London: Springer-Verlag, 1997.

[31] Hawking D., Thistlewaite P, "Relevance weighting using distance between term occurrences*", Technical Report TR-CS-96-08*, Department of Computer Science, Australian National University, 1996.

[32] Hayes J.H., "Risk reduction through requirements tracing*", Proceedings of Software Quality Week*, San Francisco, CA, 1990.

[33] Hayes J.H., Dekhtyar A., Osbourne J, "Improving requirements tracing via information retrieval", *Proceeding of the 11th International Conference on Requirements Engineering*, Monterey, CA, 2003, pp.138-145.

[34] Hayes J.H., Dekhtyar A., Sundaram S. K., Howard S, "Helping analysts trace requirements: an objective look*"*, *Proceeding of the 12th International Requirements Engineering Conference*, Kyoto, Japan, 2004, pp. 249-259.

[35] Hayes J.H., Dekhtyar A., Sundaram S, "Advancing Candidate Link Generation for Requirements Tracing: the Study of Methods", *IEEE Transactions on Software Engineering*, 32(1), 2006, pp. 4-19.

[36] He B., Ounis I, "Inferring query performance using pre-retrieval predictors*"*, *Proceedings of the Eleventh Symposium on String Processing and Information Retrieval*, Padova, Italy, 2004, pp. 43-54.

[37] He B., Ounis I, "Query Performance Prediction", *Information Systems*, 31(7), 2006, pp. 585-594.

[38] IBM Talent research group,
http://www.research.ibm.com/talent/index.html

[39] Interactive Development Environments, *Software through pictures: products and services overview*, IDE Inc, 1991.

[40] Jarke M., "Requirements Traceability", *Communications of the ACM*, 41(12), Dec. 1998, pp. 32-36.

[41] Jones K. S., van Rijsbergen C. J, "Information Retrieval Test Collections", *Journal of Documentation,* 32(2), 1976, pp. 59-75.

[42] Kaindl H, "The Missing Link in Requirements Engineering", *ACM SIGSOFT Software Engineering Notes,* 18(2), 1993, pp. 30-39.

[43] Lin J., Lin C.C., Cleland-Huang J., Settimi R., Amaya J., Bedford G, Berenbach B., Khadra O. B., Duan C., Zou X., "Poirot: a distributed tool supporting enterprise-wide traceability", *Proceeding of the 14th IEEE International Conference on Requirements Engineering*, Minneapolis, MN, 2006, pp. 356-357.

[44] Lucia A. D., Fasano F., Oliveto R., Tortora G, "ADAMS Re-Trace: a traceability recovery tool*"*, *Proceeding of the 9th European Conference on Software Maintenance and Reengineering,* Manchester, UK, 2005, pp.

32-41.

[45] Lucia A. D., Olivetto R., Totora G., "Recovering Traceability Links using Injformation Retreival Tools: a Controlled Experiment", *Proceeding of the International Symposium of the Grand Challenges for Traceability*, Lexington, Kentucky, March 2007, pp. 46-55.

[46] Marcus A., Maletic J.I., "Recovering Documentation-to-Source-Code Traceability Links using Latent Semantic Indexing", *Proceeding of the 25th IEEE International Conference on Software Engineering*, Portland, Oregon, 2003, pp. 125-137.

[47] Maletic J. I., Munson E. V., Marcus A., Nguyen T. N, "Using a hypertext model for traceability link conformance analysis", *Proceeding of the 2nd International workshop on Traceability in Emerging Forms of Software Engineering*, Montreal, CA, 2003, pp. 47-54.

[48] Marcus A., Maletic J.I., Sergeyev A., "Recovery of Traceability Links Between Software Documentation and Source Code", *International Journal of Software Engineering and Knowledge Engineering*, 15(4), October 2005, pp. 811-836.

[49] Mitra M., Buckley C., Singhal A., Cardie, "An analysis of statistical and syntactic phrases", *Proceedings of RIAO'97 Computer-Assisted Searching on the Internet*, Montreal, CA, 1997, pp. 200-214.

[50] Palmer J. D, *Traceability*, Software Requirements Engineering, Thayer R.H. and Dorfman M. (Eds), IEEE Computer Society Press, New York, 1997.

[51] Pearl J, *Probabilistic reasoning in intelligent systems: networks of plausible inference*, San Mateo, CA: Morgan Kaufmann, 1998.

[52] Pickens J., Croft W.B, "An exploratory analysis of phrases in text retrieval", *Proceedings of RIAO'2000 Content-Based Multimedia Information Access*, Paris, France, 2000, pp. 1179-1195.

[53] Pierce R, "A requirements tracing tool", *Proceedings of the Software Quality Assurance Workshop on Functional and Performance Issues*, 1978, pp. 53-60.

[54] Pohl K, "PRO-ART: Enabling Requirements Pre-Traceability", *Proceedings of the IEEE International Conference on Requirements Engineering*, 1996, pp. 76-85.

[55] PROMISE Software Engineering Repository,
http://promise.site.uottawa.ca/SERepository

[56] Ramesh B, Dhar V, "Supporting Systems Development Using Knowledge Captured During Requirements Engineering", *IEEE Transaction on Software Engineering*, 18(6), 1992, pp. 498-510.

[57] Ramesh B, "Factors Influencing Requirements Traceability Practice", *Communications of the ACM*, 41(12), 1998, pp. 37-44.

[58] Ramesh B, Jarke M, "Toward Reference Models for Requirements Traceability", *IEEE Transactions on Software Engineering*, 27(1), 2001, pp. 58-92.

[59] Rao A.G., Humphrey T., Parhizgar A., Wilson C., Pliske D, "Experiments in query processing at LEXIS-NEXIS for TREC-7", *Proceeding of the 7th Text Retrieval Conference (TREC-7)*. 1998, pp. 390-399.

[60] Rational RequisitePro, retrieved 10/29/07 from
http://www-306.ibm.com/software/awdtools/reqpro/

[61] Robertson S, Robertson J, *Mastering the Requirements Process*, Reading, MA: Addison-Wesley, 1999.

[62] Rocchio J, *The SMART Retrieval System: Experiments in Automatic Document Processing (Relevance feedback in information retrieval)*, Englewood Cliffs, NJ: Prentice-Hall, 1971.

[63] Salton G, *Automatic Information Organization and Retrieval*, New York: McGraw-Hill, 1968.

[64] Salton G, Yang C, Yu C, "A Theory of Term Importance in Automatic Text Analysis", *Journal of the American Society for Information Science*, 26(1), 1974, pp. 33-44.

[65] Salton G, Wong A, Yang C.S, "A Vector Space Model for Automatic Indexing", *Communications of the ACM*, 18(11), 1975, pp. 613-620.

[66] Salton G, McGill M, *Introduction to Modern Information Retrieval*, New York: McGraw-Hill, 1983.

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

[68] Salton G, *Automatic Text Processing: The Transformation, Analysis and Retrieval of Information by Computer*, Reading, MA: Addison-Wesley,

1989.

[69] Settimi R, Cleland-Huang J, BenKhadra O, Mody J, Lukasik W, DePalma, C, "Supporting change in evolving software systems through dynamic traces to UML", *Proceeding of the 7th IEEE International Workshop on Principles of Software Evolution*, Kyoto, Japan, 2004, pp. 49-54.

[70] Singhal A, Choi J, Hindle D, Lewis D.D, Pereira F, "AT&T at TREC-7*", Proceedings of TREC-7*, Gaithersburg, MD, 1999, pp. 186-198.

[71] Spanoudakis G, Zisman A, "Software Traceability: A Roadmap", *Handbook of Software Engineering and Knowledge Engineering*, (V. 3) S.K. Chang, World Scientific Publishing Co., 2003.

[72] Spanoudakis G., Garcez A, Zisman A. "Revising Rules to Capture Requirements Traceability Relations*", Proceeding of the 15th International Conference on Software Engineering and Knowledge Engineering (SEKE 2003)*, San Francisco CA, 2003, pp. 570-577.

[73] Spanoudakis G., Zisman A., Pérez-Miñana E., Krause P, "Rule-based Generation of Requirements Traceability Relations", *Journal of Systems and Software*, 72(2), 2004, pp. 105-127.

[74] Spence I., Probasco L, "Traceability Strategies for Managing Requirements with Use Cases", *Rational Software Technical Report*, 1999. http://www.rational.com/products/ whitepapers/

[75] Telelogic product DOORS, retrieved on 10/29/07 from http://www.telelogic.com/Products/doors/doors/index.cfm

[76] Tufis D, Mason O, "Tagging Romanian texts: a case study for QTAG, a language independent probabilistic tagger", *Proceedings of the International Conference on Language Resources & Evaluation*, Granada, Spain, 1998.

[77] Wong S.K.M, Yao Y.Y, "A Probabilistic Inference Model for Information Retrieval", *Information Systems*, 16(3), 1991, pp. 301-321.

[78] Zhai C, Lafferty J, "A study of smoothing methods for language models applied to ad hoc information retrieval", *Proceedings of SIGIR'01*, New Orleans, LA, 2001, pp. 334-342.

[79] Zou X, Settimi R., Cleland-Huang J, Miller, C S, "Supporting trace evaluation with confidence scores", *Proceedings of the Workshop on Requirements Engineering Decision Support*, Paris, France, 2005, pp. 1-6.

[80] Zou X, Settimi R., Cleland-Huang J, "Improving Automated Requirements Trace Retrieval: A Study of Term-based Enhancement Methods", Submitted to the *Empirical Software Engineering,* 2009.

[81] Deerwester S, Dumais S.T, Landauer T. K, Furnas G.W, Harshman, R.A, "Indexing by Latent Semantic Analysis", *Journal of the American Society for Information Science and Technology,* 41(6), 1990, pp. 391-407.

[82] Fellbaum C (editor), *Wordnet: An Electronic Lexical Database*, MIT Press Books, 1998.

[83] Joho H, Sanderson M, "Document Frequency and Term Specificity", *Proceeding of the 8th Recherche d'Information Assistée par Ordinateur Conference (RIAO'07),* Pittsburgh, PA, 2007.

[84] Matsuo Y, Ishisuka M, "Keyword Extraction from a Single Document using Word Co-occurrence Statistical Information", *International Journal on Artificial Intelligence Tools*, 13(1): 2004, pp. 157-169.

APPENDIX A: RETRIEVAL RESULTS IN SE450

1 Basic PN results

Project #	Precision								
	Recall =10%	Recall =20%	Recall =30%	Recall =40%	Recall =50%	Recall =60%	Recall =70%	Recall =80%	Recall =90%
1	43%	44%	48%	44%	38%	35%	30%		
2	38%	30%	27%	25%	27%	23%	18%	13%	
3	33%	40%	42%	40%	36%	33%	27%	26%	
4	42%	27%	24%	22%	21%	20%	19%		
5	16%	19%	20%	18%	13%	14%	13%	13%	
6	28%	31%	35%	36%	35%	30%	27%	22%	
7	12%	17%	16%	18%	19%	18%	18%	14%	
8	32%	40%	45%	39%	35%	34%	35%		
9	55%	56%	50%	43%	34%	32%	32%		
10	30%	23%	25%	25%	20%	18%	18%	16%	
11	34%	35%	33%	29%	27%	26%	24%	23%	
12	45%	41%	36%	35%	31%	30%	27%	25%	22%
13	40%	38%	40%	38%	34%	29%	22%		
14	22%	28%	33%	31%	27%	22%	18%		
15	53%	30%	29%	22%	22%	21%	18%	18%	18%

2 Phrasing results (algorithm T)

Project #	Precision								
	Recall =10%	Recall =20%	Recall =30%	Recall =40%	Recall =50%	Recall =60%	Recall =70%	Recall =80%	Recall =90%
1	67%	45%	48%	45%	38%	35%	30%		
2	35%	42%	33%	25%	26%	22%	19%	13%	
3	39%	43%	44%	45%	35%	32%	27%	26%	
4	50%	32%	28%	24%	22%	20%	19%		
5	14%	19%	23%	18%	13%	14%	13%	13%	
6	36%	38%	38%	38%	36%	30%	27%	22%	
7	30%	21%	21%	24%	21%	22%	18%	14%	
8	32%	40%	45%	39%	36%	34%	35%		
9	73%	54%	48%	45%	37%	32%	32%		
10	30%	33%	26%	26%	19%	18%	18%	16%	
11	39%	37%	34%	30%	27%	27%	24%	23%	
12	43%	48%	49%	34%	30%	29%	28%	25%	22%
13	45%	47%	40%	35%	34%	28%	22%		
14	41%	34%	32%	33%	28%	23%	18%		
15	57%	39%	24%	20%	21%	20%	18%	18%	18%

3 TC results (method C)

Project #	Precision								
	Recall =10%	Recall =20%	Recall =30%	Recall =40%	Recall =50%	Recall =60%	Recall =70%	Recall =80%	Recall =90%
1	63%	56%	59%	50%	37%	34%	29%		
2	60%	50%	47%	33%	31%	26%	18%	14%	
3	46%	55%	54%	51%	41%	36%	27%	26%	
4	89%	57%	25%	22%	21%	21%	20%		
5	25%	30%	25%	23%	20%	14%	14%	13%	
6	50%	58%	49%	48%	40%	35%	26%	22%	
7	29%	27%	25%	24%	24%	22%	23%	14%	
8	41%	48%	46%	47%	36%	35%	35%		
9	73%	73%	55%	45%	45%	39%	33%		
10	34%	33%	29%	26%	21%	17%	18%	16%	
11	50%	45%	41%	37%	36%	29%	26%	23%	
12	79%	55%	47%	47%	40%	34%	32%	24%	22%
13	63%	48%	51%	44%	35%	28%	22%		
14	60%	49%	33%	31%	29%	24%	18%		
15	62%	56%	37%	26%	23%	22%	21%	18%	18%

4 Results of using the original project glossary

Project #	Precision								
	Recall =10%	Recall =20%	Recall =30%	Recall =40%	Recall =50%	Recall =60%	Recall =70%	Recall =80%	Recall =90%
1	50%	42%	42%	43%	34%	33%	29%		
2	46%	45%	33%	29%	23%	21%	19%	13%	
3	41%	40%	42%	41%	32%	30%	27%	26%	
4	27%	28%	25%	23%	22%	20%	18%		
5	12%	18%	22%	20%	15%	14%	13%	13%	
6	38%	33%	36%	37%	35%	31%	26%	22%	
7	25%	21%	21%	22%	22%	19%	18%	14%	
8	34%	43%	47%	44%	41%	35%	35%		
9	69%	49%	44%	45%	35%	33%	33%		
10	27%	27%	25%	24%	20%	17%	18%	17%	
11	46%	40%	36%	31%	26%	25%	25%	24%	
12	36%	42%	45%	34%	31%	32%	28%	26%	22%
13	43%	50%	43%	36%	29%	27%	22%		
14	42%	38%	28%	29%	28%	22%	18%		
15	62%	34%	27%	23%	20%	17%	18%	12%	18%

5 Results of using the extracted set in the glossary approach

Project #	Precision								
	Recall =10%	Recall =20%	Recall =30%	Recall =40%	Recall =50%	Recall =60%	Recall =70%	Recall =80%	Recall =90%
1	71%	47%	46%	44%	37%	35%	30%		
2	43%	39%	36%	24%	25%	22%	18%	13%	
3	44%	43%	44%	43%	34%	32%	27%	26%	
4	57%	37%	29%	27%	23%	21%	19%		
5	15%	18%	22%	17%	14%	15%	13%	13%	
6	40%	41%	40%	38%	35%	30%	27%	22%	
7	26%	25%	25%	25%	22%	22%	18%	14%	
8	32%	40%	45%	39%	36%	34%	35%		
9	69%	54%	49%	47%	41%	32%	32%		
10	30%	32%	26%	26%	20%	18%	18%	16%	
11	44%	39%	33%	31%	28%	26%	25%	23%	
12	40%	48%	48%	36%	31%	30%	30%	25%	22%
13	48%	51%	40%	38%	34%	28%	22%		
14	65%	33%	31%	33%	28%	22%	18%		
15	62%	31%	29%	22%	22%	21%	19%	18%	18%

APPENDIX B: RESULTS OF APPLYING THREE ENHANCEMENT METHODS IN DIFFERENT ORDER

Order1: Apply TC first, then Phrasing and Glossary

Order2: Apply Glossary first, then Phrasing and TC

Dataset	IBS		EBT		LC	
Recall Level	**Precision**					
	Order1	**Order2**	**Order1**	**Order2**	**Order1**	**Order2**
10%	57%	69%	82%	93%	64%	64%
20%	57%	62%	75%	75%	58%	56%
30%	49%	53%	59%	63%	56%	60%
40%	48%	48%	56%	55%	54%	51%
50%	45%	47%	39%	41%	46%	48%
60%	40%	42%	34%	36%	48%	50%
70%	35%	35%	27%	29%	44%	44%
80%	30%	31%	25%	25%	40%	43%
90%	25%	25%	18%	17%	38%	38%