TEPT : A TOOL USING HEURISTIC RULES IN MACHINE LEARNING FOR DOMAIN SPECIFIC RESOURCES ACQUISITION AND TEXT PARSING

KULLAWAT WUTTISUNKORNSAKUL

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR
THE DEGREE OF MASTER OF SCIENCE (COMPUTER SCIENCE)
FACULTY OF GRADUATE STUDIES
MAHIDOL UNIVERSITY
1999

ISBN 974-663-230-2

Thesis
entitled

TEPT : A TOOL USING HEURISTIC RULES IN MACHINE LEARNING FOR DOMAIN SPECIFIC RESOURCES ACQUISITION AND TEXT PARSING

Kullawat Wuttisunkornsakul

Mr. Kullawat Wuttisunkornsakul
Candidate

Jarernsri L. Mitrpanont

Asst. Prof. Jarernsri L. Mitrpanont, Ph.D.
Major-advisor

Supachai Tangwongsan

Assoc. Prof. Supachai Tangwongsan, Ph.D.
Co-advisor

Liangchai Limlomwongse

Prof. Liangchai Limlomwongse, Ph.D.
Dean
Faculty of Graduate Studies

Supachai Tangwongsan

Assoc. Prof. Supachai Tangwongsan, Ph.D.
Chairman
Master of Science Programme in Computer Science
Faculty of Science

Thesis
entitled

TEPT : A TOOL USING HEURISTIC RULES IN MACHINE LEARNING FOR DOMAIN SPECIFIC RESOURCES ACQUISITION AND TEXT PARSING

was submitted to the Faculty of Graduate Studies, Mahidol University
for the degree of Master of Science (Computer Science)

on

October 8, 1999

Kullawat Wuttisunkornsakul

Mr. Kullawat Wuttisunkornsakul
Candidate

Jarernsri L. Mitrpanont

Asst. Prof. Jarernsri L. Mitrpanont, Ph.D.
Chairman

Supachai Tangwongsan

Assoc. Prof. Supachai Tangwongsan, Ph.D.
Member

Thanwadee Thanitsukkarn

Lect. Thanwadee Thanitsukkarn, Ph.D.
Member

Assoc. Prof. Chinda Achariyakul, Ph.D.
Member

Liangchai Limlomwongse

Prof. Liangchai Limlomwongse, Ph.D.
Dean
Faculty of Graduate Studies
Mahidol University

Pornchai Matangkasombut

Prof.Pornchai Matangkasombut, Ph.D., M.D.
Dean
Faculty of Science
Mahidol University

ACKNOWLEDGEMENTS

I would like to express my gratitude to those, who helped me and contributed to the completion of this thesis.

I am greatly indebted to my thesis advisor, Dr. Jarernsri L. Mitrpanont, for her support, concern, encouragement and valuable advice throughout my period of study. Also to my thesis supervisory committee members, Dr. Supachai Tangwongsan, Dr. Chinda Achariyakul, and Dr. Thanwadee Thanitsukkarn, for their constructive suggestions.

Many sincere thanks to the Department of Computer Science of Mahidol University for providing the knowledge in building and completion of this thesis.

Finally, I am deeply indebted to my parents, my wife and my son for their love, understanding and encouragement during my study.

Kullawat Wuttisunkornsakul

3737096 SCCS/M : MAJOR : COMPUTER SCIENCE ;
M.Sc. (COMPUTER SCIENCE)
KEY WORDS : HEURISTIC RULE, TEXT PARSING, DOMAIN LEXICON, MACHINE LEARNING

KULLAWAT WUTTISUNKORNSAKUL : TEPT : A TOOL USING HEURISTIC RULES IN MACHINE LEARNING FOR DOMAIN SPECIFIC RESOURCES ACQUISITION AND TEXT PARSING. THESIS ADVISORS : JARERNSRI L. MITRPANONT, Ph.D., SUPACHAI TANGWONGSAN, Ph.D. 67 P. ISBN 974-663-230-2

This thesis studies another approach of Natural Language Processing (NLP) supporting text parsing via Machine Learning controlled by a set of heuristic rules. Information flooding on the Internet and lack of a domain specific parsing tool, require domain specialists to waste time manually selecting domain information. Lexicons are the major resource used to select domain documents. Specialists need a suitable and efficient tool to extract domain keywords such as technical terms that are not in a standard dictionary. In addition, this tool should be easily ported for use in any new domain. This study proposes a Text Efficient Parsing Tool (TEPT) as a new NLP tool to assist domain specialists in extracting keywords from domain documents. The TEPT system in this study applies heuristic rules and machine learning techniques to acquire domain specific resources and lexicons which are essential for parsing. This Dynamic Learnable Lexicon Feature is one of the most outstanding feature of the TEPT. It can be used to self-develop a set of lexicons in any trained domain. Using this TEPT, essential information or domain keywords can be extracted from parsed text via the domain lexicon. Study results demonstrate that after training TEPT it can develop lexicons in Computer Science domain. These results suggest it can be used effectively to extract keywords from other specific domains.

3737096 SCCS/M : สาขาวิชา : วิทยาการคอมพิวเตอร์ ; วท.ม. (วิทยาการคอมพิวเตอร์)

กุลวัฒน์ วุฒิสารกรสกุล : TEPT : เครื่องมือที่ใช้ Heuristic Rules ในการเรียนรู้ด้วยเครื่องเพื่อการสร้างปัจจัยหลักในการวิเคราะห์ไวยากรณ์ภาษาและใช้งานใน Domain เฉพาะที่ผู้ใช้สนใจ (TEPT : A TOOL USING HEURISTIC RULES IN MACHINE LEARNING FOR DOMAIN SPECIFIC RESOURCES ACQUISITION AND TEXT PARSING). คณะกรรมการควบคุมวิทยานิพนธ์ : เจริญศรี มิตรภานนท์, Ph.D., ศุภชัย ตั้งวงศ์ศานต์, Ph.D. 67 หน้า. ISBN 974-663-230-2

วิทยานิพนธ์ฉบับนี้ได้ศึกษาวิจัยถึงการใช้แนวทางของ NLP ในการทำ Parsing โดยใช้เทคนิคการเรียนรู้ด้วยเครื่องร่วมกับ Heuristic Rules เนื่องจากการที่ข้อมูลสารสนเทศจำนวนมหาศาลมีการไหลเวียนบนเครือข่าย Internet และการขาดแคลนเครื่องมือที่ใช้ในการคัดเลือกเอกสารที่ตรงตาม Domain เฉพาะสาขา ทำให้ Specialists ในสาขาเฉพาะด้านต้องเสียเวลาไปในการคัดเลือกข้อมูลสารสนเทศที่ตรงกับ Domain เฉพาะของตน สิ่งเหล่านี้ได้ทำให้ Specialists ในแต่ละ Domain มีความต้องการเครื่องมือที่เหมาะสมและมีประสิทธิภาพเพื่อช่วยในการค้นหา Domain Keywords ในเอกสารโดยเฉพาะอย่างยิ่ง Domain เฉพาะที่เกี่ยวข้องกับ Keywords ประเภทศัพท์เทคนิคต่างๆซึ่งมิได้ปรากฏอยู่ในพจนานุกรมมาตรฐานทั่วๆไป นอกจากนั้นเครื่องมือนี้ควรมีความสามารถในการนำไปใช้กับ Domains อื่นได้ง่ายด้วย การศึกษาวิจัยนี้ได้นำเสนอเครื่องมือวิเคราะห์ไวยากรณ์ภาษาที่มีประสิทธิภาพในเอกสารต่างๆที่เรียกว่า TEPT หรือ Text Efficient Parsing Tool โดย TEPT ประยุกต์ใช้เทคนิค Heuristic Rules และ Machine Learning ในการวิเคราะห์ไวยากรณ์ภาษาในเอกสาร และสามารถสร้างพจนานุกรมซึ่งจำเป็นต่อการวิเคราะห์ไวยากรณ์ภาษาได้เองโดยอัตโนมัติ ซึ่งถือเป็นคุณลักษณะที่สำคัญของ TEPT การสร้างพจนานุกรมที่มีการหมุนเวียนเคลื่อนไหวของคำศัพท์อยู่ตลอดเวลาและสามารถเรียนรู้คำศัพท์ใหม่ๆได้ด้วยตัวเองนี้มีประโยชน์ต่อการคัดเลือกข้อความสำคัญในเอกสารในแต่ละ Domain ที่ผู้ใช้สนใจ การใช้ Heuristic Rules เพื่อควบคุมขั้นตอนของ Machine Learning ในการสร้างพจนานุกรมก็ดี หรือในการทำ Text Parsing ก็ดี วิธีนี้ทำให้ได้พจนานุกรมในแต่ละหัวข้อเรื่องที่ผู้ใช้สนใจโดยขึ้นอยู่กับเนื้อหาในเอกสารเหล่านั้นเป็นสำคัญ และท้ายที่สุดข้อมูลสารสนเทศที่สำคัญและตรงกับ Domain ในเอกสารก็จะถูกคัดเลือกออกมา จากผลการวิจัยแสดงให้เห็นว่า TEPT สามารถคัดเลือก Keywords ใน Computer Science Domain ได้อย่างมีประสิทธิภาพภายหลังจากผ่านการสร้างพจนานุกรมมาแล้ว

CONTENTS

Page

Page

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

CHAPTER I

INTRODUCTION

Proliferation of electronic document on computer networks has driven the research to develop the method for information management. No average human can read, process, and understand a large amount of these data every day. Moreover, they even spend too much time to screen relevant text from this gigantic size of data. As more information becomes available, the users are flooded with more knowledge.

Although it is difficult to process all of these documents, there is an alternative in selecting information, i.e., to narrow down to topics that users interest or *domain specific*. With the latter behavior, users still have to manually select the information by themselves. Yet, it is time-wasting for any specialist or knowledge worker of that specific domain to go through all the available documents. Therefore, they need a suitable and efficient tool to extract the domain keywords from any documents. The domain specific information depends on domain specific keywords. These keywords help the users to make decision in selecting domain documents. The problems of high volume, extra time to process data, and a tool suitable and efficient for domain specific tasks have challenged many researchers to explore new information management tools that can help classify or select the information that is relevant to their domain interests.

With the time constraint and the expertise in each specific domain, a tool that is suitable for this task must be self-learning and dynamic. In addition, since the domain specific keywords play a significant role in information content, a tool for finding these domain keywords will be appropriate to improve users' selecting ability, especially, for domain specialists tasks. In general, these tasks involve with keywords that are mainly technical terms not in standard dictionary such as TCP/IP.

From literature review, the most common information management systems are such as information retrieval (IR), information extraction (IE) and Text Categorization (TC). These previous works are mainly concerned with high precision results in information management. Many studies use natural language processing technique in their parsing systems for high performance. Most of them emphasize on the correctness of parsing results but not in resource acquisition, such as lexicon and grammar patterns, which are little concerned [3][4][7][13].

In this thesis, we believe that more work should be done in the area of new resources acquisition to catch up with rapid emergent of many new text domains browsed on the internet. In addition, the parsing efficiency also depends on these resources. With this idea, we propose a new parsing tool called a text efficient parsing

tool (TEPT), using two techniques: heuristic rules and machine learning. Based on NLP, TEPT is proposed as a new efficient parsing tool in information management for any domain specific. This research emphasizes on the approach in resources acquisition for parsing system. Heuristic rules play a significant role in TEPT's automated knowledge acquisition such as lexicon, grammar patterns, and also help controlling the correctness of parsing results. Furthermore, Machine Learning is used to support the users in acquiring knowledge from training texts.

The significant aspect of TEPT relies on its learnable feature in domain specific lexicon acquisition. Specifically, TEPT algorithms are domain independent, and the domain specific lexicon can be acquired automatically, given an appropriate training corpus. In training phase, verb lexicon is learnt and automated from the appropriate corpus. It is then used to develop the noun phrase and grammar patterns. Thus, the system is trainable and portable to any new domains. This feature is called *dynamic lexicon*. The content in dynamic lexicon, as well as, size of lexicon depends on each user's interested topic. A complete dynamic lexicon from any domain topic will cover all topics in the world. These called a domain independent lexicon.

Our research also emphasizes on high precision text parsing. In many real-world applications, users are happy to receive a smaller number of text documents with high accuracy for parsing rather than a bigger one with lower accuracy. In contrast, a tradeoff between processing as many texts as possible and processing only texts within each domain are significant. As a result, the high precision domain specific lexicon helps filtering the non-relevant texts before parsing which is another advantage of TEPT. Since TEPT performs only 100% tagging words in a sentence based on words in domain lexicon, it will produce output within that specific domain with a minimum ambiguity. This implies that TEPT will produce a smaller size of relevant output with higher precision. Consequently, it will automatically help a user in reducing parsing time spent as well.

The thesis consists of seven chapters. The chapters' organization is as the following. Chapter 1 presents the overview of information management and TEPT main ideas. Chapter 2 studies on the natural language processing with some of the previous approaches on how to solve the problem in information management. Chapter 3 describes the need, motivation and objectives in TEPT approach. Chapter 4 presents TEPT architecture and approaches in automated knowledge acquisition and parsing algorithm. Chapter 5 illustrates the design of TEPT prototype. Chapter 6 presents the experimental results of the prototype in heuristic rules evaluation and significant phrases extracted from texts. Finally, chapter 7 contains the conclusions of this present work.

CHAPTER II

LITERATURE REVIEW

2.1 Information Management

Much of the information circulating in the world consists of written, natural language text such as textbooks, newspapers, journal articles, and other published materials. After the Internet is adopted, there are huge information from over the world to investigate. All of these information are available in machine-readable form and can be stored, reproduced and transmitted from place to place on computer networks. With the advent of these information, there are a radical change in the information consumption behavior of the average human being.

There are more data in electronic form than ever before. A large portion of time is spent processing on the voluminous information circulating around us, and much time is wasted on processing unuseful information. Therefore, some important information will be lost and ignored. The problem of high volume of data and time to process data has challenged many researchers in attempting to develop the techniques that used to manage these information. A variety of applications of natural language processing (NLP) are developed to support their goals. Information management such as Information Retrieval, Information Extraction and Text Categorization are some good examples in nowadays.

2.1.1 Information Retrieval (IR)

Information retrieval (IR) consists of the process of retrieving information that matches to what a user wants. Given a set of documents and a user query, IR does the task of finding the relevant documents. Traditionally, a user types in a set of keywords and the system retrieves a set of matching documents. Almost all IR systems use the simple natural language techniques to process a document. Each sentence in data text is broken down into many words. Some common words or stoplist (and, of, or, but, the, etc.) which considered no indexing value are deleted. Using a suffix stripping routine or word stemming to reduce the remaining words to word stem form. These words are called a set of keywords. For retrieving documents, the system uses a technique called boolean combinations of these keywords to search in the documents [18].

There are many different approaches that use more sophisticated natural language techniques to increase the accuracy in information

retrieval, for example, linguistic processing approaches and knowledge-based approaches [19]. The linguistic processing approach uses syntactic analysis to find out words or phrases from a document to form a set of keywords. The knowledge-based approach uses a semantic network structure for retrieving documents.

2.1.2 Text Categorization

Text categorization is an information task in which one or more category groups are assigned to a document. It performs a sorting function on many text documents and separates them into several predefined groups.

Text categorization uses word distribution [1] in all documents to find a set of keywords which represent each category. These keywords called domain specific keywords are used in classification process. The system scans all texts and looks for these domain-specific keywords to classify the text as relevant or not. They do not use the entire representation of a text to decide whether the text is relevant.

Traditional approaches to information categorization use keyword searches and statistical techniques to classify relevant documents. However, these approaches have well-known limitations because one word can have multiple meaning, and complex linguistic patterns cannot be recognized by keywords alone. So, it is difficult to find relevant document with high correctness classification by using words alone [14].

2.1.3 Information Extraction (IE)

Information extraction (IE) has the goal of extracting information from a document. IR systems collect the texts while IE starts with a collection of such texts, and processes them. IE uses text skimming technique that selectively processes text that is relevant to a domain. With this technique, IE looks for the significant keywords in text and extracts some portions of a text. The portions without the keywords will be ignored. IE helps the extraction faster because it skips the text that are irrelevant or too complex.

IE extracts information from text by matching each sentence with extraction rules or extraction patterns. These rules or patterns are indexed with keywords which are mainly used in matching sentence [2].

The comparison of three approaches are summarized in table 2.1 below. We compare them in general factors.

Table 2.1 depicts the comparison of IR, IE and TC

	IR	IE	TC
Technique used	NLP	NLP	NLP
Key searched	words or phrases	pattern matching	words or phrases
Time processed	take time	save time	save time

2.2 Information Management Techniques

2.2.1 Word-based Technique

The early information management systems, such as IR and TC, used word-based technique [17]. This technique was based on the extraction of a set of single term from document texts and used them as keywords for searching or classifying. The step in selecting keywords was simple. First, the document was broken down into sentences. The sentences were decomposed into a set of words and then discarded the common words or stopword lists from group set. By using stemming algorithm, each word is reduced into its basic form. Finally, the remaining words were assumed as a set of keywords.

Unfortunately, it was found that word-based technique has many limitations to achieve high accuracy. Synonymy is a well-known problem for this technique. For example, the words "make", "manufacture", and "produce" all refer to the concept of production. Polysemy is another problem that some words can have multiple meaning. For example, the word "post" can refer to the name of a newspaper, a vertical support in carpentry, entering a transaction in accounting, or sending a message in computer. Some words are good indexing terms only in specific phrases. For example, the phrase "passed away" means that someone died, but the word "passed" and "away", used independently, are not generally associated with dying. These words can lead to false interpretation.

2.2.2 Word Phrases Technique

To overcome the limitation of single words, the word phrases or word grouping approach was introduced [10],[22]. This approach used a set of delimeters consisting of about 300 stopwords and punctuation symbols and sequences of two to four adjacent words enclosed by delimeters are selected as phrases.

2.2.3 NLP-based Technique

In general, word-based and word phrases techniques can be used in IR and TC but in IE, these two techniques are not complex enough because IE uses extraction patterns with keywords to extract the

significant portion in text. These extraction patterns, based on linguistic pattern, show how their components in the pattern relate to each other. Only words or phrases cannot indicate these relationships between them.

Because documents and texts are natural language constructs which consisted of many sentences, linguistic technique [18] used syntactic analysis to assign syntactic tags to each word of the input sentence and combined words into group phrases. The output presented their relationships in the structure of grammar language.

However, natural language processing uses an expensive cost of computational resources. There are many approaches used to overcome NLP problems such as syntactic and semantic problem. But most of them need the highly skilled ability of expert to solve the problem.

2.3 Ideal NLP System

The above three applications (IR, TC, and IE) have the same common characteristics of text processing technique, i.e., NLP-based technique. The demands of natural language research have led to the development of new NLP techniques in large-scale text processing. The main factors involve in ideal text processing techniques require accuracy, speed and ease of use [8].

Accuracy

The performance of the text analysis can be computed in two main measure terms. The first term is "recall" which means how much correct information was selected. The second term is "precision" which means how much of the information selected was correct. The results of high recall and high precision are needed.

Speed

As a number of text increases, the speed of text analysis becomes a severe bottleneck. The need for fast text analysis will reduce the cost of producing tasks to acceptable levels.

Ease of use

Text analysis tool should provide a user interface that allows easy access to the results.

Within these three properties, the accuracy is the most important aspect in any NLP-based text processing system. Many researchers have developed their tasks in order to get high accuracy.

2.4 Trend of Natural Language Processing

Research and development in NLP have grown over the past few years, and here are some changes for the new research [8]:

From full parsing to partial parsing

Partial parsing refers to a sentence analysis that does not require a full syntactic parse of a sentence. A full syntactic analysis of every sentence in every text is too computational. Most of the effort in successful text processing systems now goes into compensation for the limitations of parsing rather than producing more complete parses.

From sentences to text

Like a human reader, he looks for the focus phrases or words by skimming all over the texts. He can quickly and easily pick out them. The relationship among these focus phrases or words can be easily understood.

From raw text to marked-up text

Marked-up texts prevent unnecessary parsing, and can easily highlight the essential parts in the text. It is effective to reduce the number of parsing decisions that must be made and to help guide those decisions.

From expert to non-expert

Formal text analysis systems require the ability of linguistic experts to examine the correctness of the system. Sentence analysis itself involves constraint satisfaction from multiple knowledge sources. Can non-language specialist do this task by himself?

The important motivation in these new changes is to reduce the complexity in NLP tasks and gain the high accuracy. Moreover, the researchers have expected that it should be developed by non-linguistic users rather than by linguistic experts.

2.5 Syntactic Processing

This section will begin with an overview of the natural language processing and some of the approaches that have been tried. Before we go into detail on the several factors of the natural language processing, it is useful to survey all of them and see how they work together. The major NLP techniques

[15],[20],[21] can be divided into three groups: Parts of Speech, Grammars, Parsers and Parsing techniques.

2.5.1 Parts of Speech (POS)

Every sentence composes of a combination of words. Many words are put together in order to make sense. Each word in a sentence is a basic unit and has its syntactic categories to form a meaningful sentence. The important categories for POS are noun, pronoun, verb, adjective, adverb, conjunction, and preposition. This technique assigns POS categories to each word that are presented in the text.

2.5.2 Grammars

The grammars are a set of rules that show how the POS tagged words relate to each other in a sentence. The grammar is represented by a set of production rules as illustrated in Figure 2.1.

Sentence	-> Noun-Phrase Verb-Phrase
Noun-Phrase	-> Determiner Noun
Verb-Phrase	-> Verb Noun-Phrase

Figure 2.1: A simple grammar

2.5.3 Parser and Parsing techniques

The key element in a natural language system is a parser. It maps the words from the sentence into a structure called a parse tree. This process is called parsing. The term "parsing" refers to the identification of different components presented in the sentence. By using rules of the grammar and POS of each words in the sentence, the parsing process compares them against the input sentence to produce parsed structures. An example of a parse tree is demonstrated in Figure 2.2.

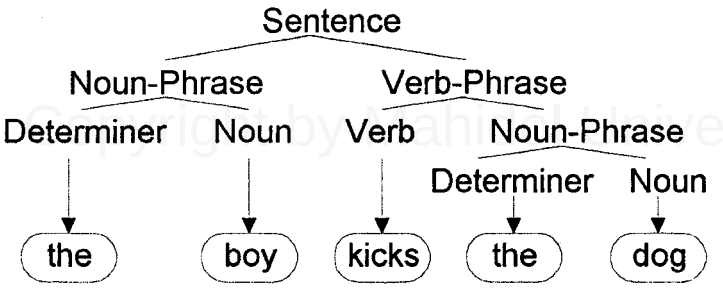

Figure 2.2 : A parse tree for a sentence "the boy kicks the dog"

In general, parsing techniques are classified into two methods: top-down and bottom-up parsing. Top-down parsing is an approach to match a sentence forward with grammar rules and generate a parse tree for output while bottom-up parsing matches backward.

According to [6], the early parsing systems have two major categories: Augmented Transition Networks (ATN) and Context Free Grammars (CFG).

- Augmented Transition Networks (ATN)

The ATN was used in early AI systems. ATN provides a well-defined framework for mapping a sentence into syntactic or semantic relations. Based on the finite state machine model, an ATN is a recursive transition network (RTN) in which subgraphs can be labeled and used as transition symbols.

ATN begins with the start symbol and applies the grammar rules forward until the symbols at the terminals of the tree correspond to the components of the sentence being parsed. In general, its processing features look like top-down parsing.

Although ATNs are powerful and can be made reasonably efficient, but ATN grammars are often difficult to maintain.

- Context Free Grammars (CFG)

Context Free Grammars were used in the fields of linguistics and computer science such as compiler design. The formal properties of CFG have been extensively studied because of their use in the design of computer languages and the construction of parsers for CFG is well understood.

CFG begins with the sentence to be parsed and the grammar rules are applied backward until a single tree whose terminals are the words of the sentence and whose top node is the start symbol has been produced. In general, its processing features look like bottom-up parsing.

2.6 Lexicon Design

Because meaningful sentences are composed of meaningful words; any system, that hopes to process natural languages as people do, must have information about words and their meanings. This information is traditionally provided through dictionaries or lexicon. Most NLP systems need to know the part of speech of the words in order to parse them appropriately.

Almost all NLP projects cannot function without consulting a lexicon. In order to perform the syntactic processing, the parser needs a dictionary. This dictionary is called the lexicon. The parser and the lexicon work together to parse a sentence and create the parse tree. The parser searches through the lexicon, compares each word in the sentence, and returns part of speech of that word.

A lexicon is a large knowledge base that contains common knowledge about words and phrases. It can be obtained in three different ways: machine-readable dictionary, manual creation dictionary, and domain-specific semantic dictionary.

- The Machine-Readable Dictionary

  Some systems use the global lexicons from the machine-readable version of Longman's Dictionary of Contemporary English (LDOCE), a British-produced learner's dictionary, which contains 35,899 headwords and 53,838 senses [1].

  The use of machine-readable dictionary is an easy way to solve the problem but it becomes a difficulty when there are some words that do not appear in standard machine-readable dictionary.

- The Manual Creation Dictionary

  In CONSTRUE-TIS system [1], it uses a manually developed lexicon of words and phrases. Some systems use a small general lexicon. For example, SCISOR [7] uses a lexicon of about 10,000 word roots, with about 75 affixes (prefixes and suffixes) that can derive variations from those roots to tag words, and a specific domain pattern knowledge.

  It is not easy to develop the manual creation dictionary because the design and size of dictionary are major problems. Moreover, the developing time is another factor that should be considered.

- Domain-specific Semantic Dictionary

  It is essential to have at least a few thousand common words to avoid building a new lexicon. Much of the new work in large-scale natural language processing concentrates on how to acquire enough lexical knowledge for any domain-specific lexicon. AutoSlog [16] is a dictionary construction system that creates concept nodes automatically using heuristic rules. The concept node in dictionary could handle sentence analysis with respect to interactions between syntax and semantics. Each of the following examples is followed by a rule derived

automatically for inclusion in Latin American terrorism in the dictionary [5]:

1. Rule:
   The direct object of "robbed" (active voice) is the victim of a robbery.
   A group of armed individuals wearing ski masks **robbed** a businessman.

2. Rule:
   The subject of "disappeared" (active voice) is the victim of a kidnapping.
   The Assistant Secretary **disappeared** on Jan. 12.

3. Rule:
   The subject of "placed" (passive voice) is the instrument of a bombing.
   A bomb was **placed** in the parking lot of Government House.

4. Rule:
   The object of "in" after traveling (active voice) is the target of an attack.
   The group was **traveling in** a four-wheel-drive vehicle.

5. Rule:
   The subject of "hurled" (passive voice) is the instrument of an attack.
   Reported that dynamite sticks were **hurled** from a car.

2.7 Building Grammars

As described above, the grammar is represented by a set of production rules. These rules are vary in patterns. When a parser applies grammar rules to a sentence, it performs two major operations:

- Matching (of sentence constituents to grammar rules)
- Building structure (corresponding to the result of combining constituents)

There are many different techniques of parsing systems in natural language systems. For example, ATN or Augmented Transition Networks [15] is a top-down parsing procedure described as the transition from a start state to a final state in a transition network that corresponds to a grammar of English.

Some systems use PROLOG language programming to write grammar rules as clauses and interpreter.

Two main approaches have been used to provide the knowledge for grammar rules.

1) Knowledge-engineered technique

This approach uses the knowledge engineers to act as the connection between the computer and the expert. These engineers use the knowledge acquisition techniques to acquire the grammar rule knowledges from the expert and develop the representations and transfer them into the computer language.

Advantages:

- The users can select the appropriate system for their tasks.

Disadvantages:

- The users cannot modify all what they want.

2) Machine learning technique

This approach allows a user to prepare the training data sets to make grammatical rules. With this technique, we will not worry about knowledge-engineering bottlenecks. Having human experts do this job is very expensive. Having non-experts do it is cheaper. In contrast, having machines do the job is even cheaper.

Advantages:

- The user can easily manage the system in order to support their goals.
- It is easy to develop the system, while the time and cost are minimized.

Disadvantages:

- With this technique, the system requires large amount of training sentences.
- In some complex case, it is essential to use experts to solve that case.

2.8 Lexical Ambiguity

Lexical ambiguity is a problem in natural language processing. It provides an inaccurate result in text parsing or text interpretation. To receive an accurate result, the system should reduce this effect.

According to [11], the literature generally divides lexical ambiguity into two types: syntactic and semantic. Syntactic ambiguity refers to differences in syntactic category (e.g., *play* can occur as either a noun or verb). Semantic ambiguity refers to differences in meaning (e.g., a word can have several meaning).

The way to reduce ambiguity is presented by comparing relative positions of words with respect to each other in the range of two to five words. For example, the words appear at positions -1, -2, +1, and +2 where the negative represents left position and positive represents right position.

2.9 Machine Learning Technique

Machine learning [12] is in a field of Artificial Intelligence. It is the study of computational methods for solving the problem or improving performance by acquiring knowledge from the expert's experience. It is the dream system of AI. Machine learning aims to provide increasing levels of automation in the knowledge engineering process, replacing much time-consuming human activity with automatic techniques that improve accuracy or efficiency by discovering and exploiting regularities in training data. In this research, we used machine learning in order to apply to natural language problems, including part of speech tagging, prepositional phrase attachment disambiguation, and syntactic parsing.

Most recent research in natural language processing has explored a learnable system such as, trainable part of speech taggers, trainable parsing system, and trainable lexicon. Here are some approaches.

2.9.1 Text Annotation as Source of Knowledges

The systems have used an annotated training text or some form of annotated input with syntactic or semantic tags for training system. Generating an annotated sentence is a significant undertaking, both in time and difficulty. Some system has created the grammar by reading the production rules directly from hand-parsed sentences.

2.9.2 Get Knowledges from Supervised Learning by Users

To reduce the time required to annotate training sentences, we apply this approach to overcome the problem. This approach acquires knowledges with the help of a teacher. The term "supervised" originates from the fact that the output nodes are provided by an external "teacher." Supervised learning uses a set of input data for which the appropriate (desired) outputs are known. The system learns from the examples that a teacher supplies. The teacher is an external evaluator to judge whether or not the output is correct for each training example. The set of these

training set contains elements which consist of paired values of the independent (input) variable and the dependent (output) variable. Good result requires training on a corpus of several thousand examples from the domain.

For example, "the TreeBanker is a graphical tool for the supervised training involved in domain customization of the disambiguation component of a speech- or language-understanding system. It presents a user, who need not be a system expert, with a range of properties that distinguish competing analyses for an utterance and that are relatively easy to judge. This allows training on a corpus to be completed in far less time, and with far less expertise, than would be needed if analyses were inspected directly: it becomes possible for a corpus of about 20,000 sentences of the complexity of those in the ATIS corpus to be judged in around three weeks of work by a linguistically aware non-expert." [3]

Advantages:

- It is easy for the user to judge for the correctness of the result.

Disadvantages:

- A user only needs to provide sample texts, and spend more time judging the results.
- It is not easy for the user to select the overall grammar patterns in training.

2.10 Existing Information Management Tool

Many researchers have developed several tools to process information in text. Each of these tools has its particular features. In this section, four well-known tools: Circus, ENGCG, Scisor, and TreeBanker are explored.

1. Circus

Circus [13] is a sentence analysis parser system presented by Lehnert at Massachusetts University. It is used in IE task. First, it recognizes clause boundaries and syntactic constituents in sentence, such as subject, verb, direct object, and prepositional phrases of each clause, and scans for a specific word (called trigger word). Then, Circus compares the fragments with pre-defined pattern of trigger word by using semantic-feature tagger for identifying noun phrases. Finally, Circus returns the extracted information in slot forms.

2. ENGCG

ENGCG [4] is a Constraint Grammar Parser of English. ENGCG is based on the Constraint Grammar framework originally proposed by Prof. Fred Karlsson, Head of the Research Unit for Computational Linguistics, Department of General Linguistics at the Univeristy of Helsinki. ENGCG uses 200 heuristic constraints for tagging ambiguous words and 1,200 grammar-based constraints for determining ambiguous syntactic functions.

3. Scisor

Scisor [7] is a prototype filtering tool presented by Jacobs and Rau at GE Research and Development Center. It operates on financial news by using the integrated bottom-up and top-down techniques with semantic tagger (called conceptual structures) for parsing texts.

4. TreeBanker

The TreeBanker [3] is a tool for the supervised training of disambiguation in natural language speech sentence. It applies syntactic and semantic rules to create ambiguous parsing output and allows the non-expert user to judge for the correct parsing analysis.

2.10.1 Comparison

To make a comparison of the four tools mentioned above, the following four comparative measures are considered: lexicon, syntax, parsing technique and system learning.

Table 2.2 shows the comparison of parsing systems

Parsing Systems	Lexicon	Syntax	Parsing Technique	System Learning
Circus	MRD & Domain	Pre-defined grammar	Bottom-up and Top-down	N
ENGCG	MRD	Pre-defined grammar & Heuristic Rules	Bottom-up and Top-down	N
Scisor	Manual & Domain	Pre-defined grammar	Bottom-up and Top-down	N
TreeBanker	Manual & Domain	Pre-defined grammar	Bottom-up and Top-down	Y

2.10.2 Issues

The significant issues of these existing tools can be summarized as follows:

1. **Domain specific**

   Circus, Scisor, and TreeBanker work in each domain specific. Circus used in terrorism domain, while Scisor and TreeBanker used in financial domain and ATIS (Air Travel Inquiry System) domain respectively.

2. **Resources Acquisition (Lexicon and Grammar)**

   The resources for parsing come from any ways. Circus gains lexicon from MRD and uses domain lexicon with pre-defined grammar to parse sentence. ENGCG uses MRD with pre-defined grammar and reduces ambiguation by using heuristic rules. Scisor and TreeBanker build manual domain lexicon and works with pre-defined grammar. None of these tools concern about resources acquisition. They emphasize on the correctness of parsing output.

3. **System Learning**

   TreeBanker allows the non-expert user to judge for the disambiguation of correct parsing analysis and collects learnable results into knowledge of each ambiguous word.

2.11 Heuristic Method

We apply heuristic method for learning natural language processing tasks. Heuristic method is the approach of employing heuristic rules to arrive at feasible and "good enough" solutions to some complex problems. "Good enough" is usually measured in the range of 90-99.9 percent of the solution. Heuristic rules are developed through experience and judgement. It is often called rules of thumb.

The major advantages of heuristics are that they are simpler to understand and easier to implement. They save time in solving the problems. With this benefit, we expect that the heuristic method can be used in machine learning, and can overcome the language problem.

CHAPTER III

PROBLEM STATEMENT

3.1 The Needed Domain Specific Resources for Specialists in each Domain

Previous works have shown that several existing parsing tools work only in their domain specific field, for example, Circus used in terrorism domain while Scisor and TreeBanker used in financial domain and ATIS (Air Travel Inquiry System) domain respectively. These tools work in static manner and used in some particular domains with a certain difficulty to port to other domains. Expert, specialist, or knowledge worker in many domain specific such as Computer Science, Engineering, have no tools to use in their tasks.

With the lack of domain specific parsing tools, it is time-wasting for specialists or knowledge workers to manually select domain information through a huge volume of knowledge circulating on the Internet everyday. They are confronting with information flooding problem. To reduce the time spent, the specialists need suitable and efficient tool to manage these information by extracting domain keywords or technical domain terms from any documents by themselves.

3.2 The Need for Efficient NLP Parsing Tool to Extract Domain Keywords

As depicted in figure 3.1, the keywords or significant phrases in the document are the important basic components for these tasks. IR, IE and TC need the efficient parsing tool in text analysis to find the text representing keywords. The high accuracy result in selecting keywords is important for successful task operation and achieving the high performance in text analysis. There are many questions about the method of how to recognize the significant words with high accuracy. The important of how to extract significant words with high accuracy and the difficulty in extracting them will be considered particularly under the various problems, such as the complex structure of sentence, and the minimum expert involvement.

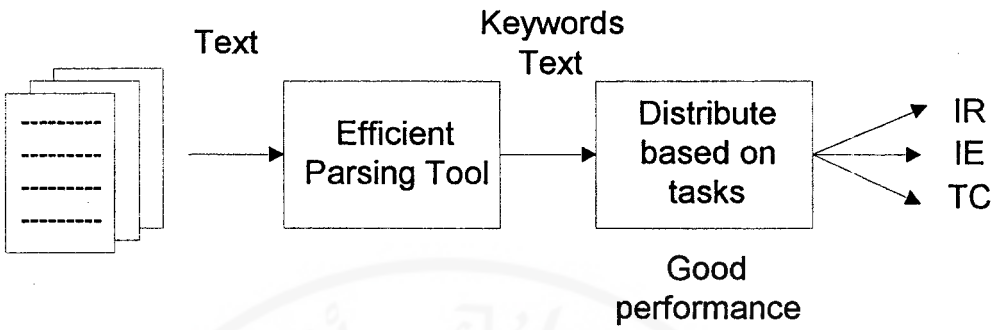

Figure 3.1 illustrates the need for NLP parsing tool

Natural language processing plays a significant role in information processing systems. NLP systems require information from several different resources to analyze text correctly. These resources include a lexicon, a grammar, and a parsing technique. Natural Language parsing is dependent on the lexicon and the grammar rules. The parser uses the lexicon to identify Part-of-Speech of words in the sentence and matches them with the grammar rules to produce parse tree.

Almost all NLP systems cannot function without consulting a lexicon. The use of machine-readable dictionary is an easy way but it becomes a difficulty when there are some words that do not appear in standard machine-readable dictionary; the domain specific terminology, per se. On the other hand, the use of manual creation dictionary is another way but the design and size of the dictionary are the important problem. As well as the lexicon problem, acquiring for grammar rules is still needed from experts to develop and represent them in the computer language.

3.3 Research Motivation and Objectives

With the problem above, there will be a motivation in developing the tool that is more appropriate to any domain specialists in extracting domain specific keywords or technical domain terms from domain information. The domain specific lexicon will help the specialists select domain documents and reduce time in finding new domain keywords.

This tool should be automated and learnable for resources acquisition in any domain specific. The advantage of this feature reduces time in knowledge-engineering bottleneck of resources acquisition from expert. The other significant feature is dynamic domain lexicon. This dynamic feature helps the tool portable to any new domain based on the specialist interests. We believe that automated resources acquisition and dynamic domain lexicon make our tool tool portable to any new domain based on the specialist interests. We

believe that automated resources acquisition and dynamic domain lexicon make our tool reasonable to NLP tasks when compare with other parsing tool.

Moreover, we also need a tool that both reliably predicts correctness of analyses and allows the specialists to inspect or judge for the correct analysis by themselves. Finally, we hope that it can be used as the extension of domain expert systems for its learning feature, particularly in the specific domain lexicon builder.

In this thesis, a model of method for learning natural language processing and parsing significant phrases in the text is presented.

The objectives of this thesis are to:

1) propose a model of a new natural language processing parsing tool architecture for specialist in each domain as option to solve the problem of Information Flooding;

2) use the heuristic rule concepts with machine learning techniques in designing automated learnable lexicon, grammar, and parsing based on the proposed model;

3) develop the prototype to demonstrate the idea by using this approach;

4) evaluate prototype and parsing results to show that the system is appropriate and robust to domain specific task for any domain specialists.

CHAPTER IV

TEXT EFFICIENT PARSING TOOL ARCHITECTURE

Previous studies have shown that the important success factors in information management are the need for efficient text parsing tool and resources used in text parsing. Natural language processing systems require information from several different resources to parse text correctly. These resources include a lexicon, a grammar, and a parsing technique. The major issue in this thesis is concentrated on how to develop these resources automatically with a minimum expert involvement.

In this thesis, we apply the heuristic rules and machine learning technique to natural language processing parsing tool and resources acquiring. These two techniques involve in the acquisition of parsing knowledge from text data. The heuristic rules assist a user by assigning the knowledge rules in language including lexicon, grammar, and syntactic parsing. These rules identify part-of-speech of words or grammar patterns automatically from training text. The machine learning techniques allow the user to get or extract knowledge from the pre-defined heuristic rules into lexicon resources. A detail will be described in the following section.

4.1 Text Efficient Parsing Tool (TEPT) Architecture

This chapter outlines the proposed model for the Text Efficient Parsing Tool or simply called TEPT. We propose the following five major components to be essential for solving the problems: Lexicon Builder, Text Pre-Processor, Control Mechanism, Adapted Lexicon Builder and Heuristic Rules. The overall system is illustrated in Figure 4.1.

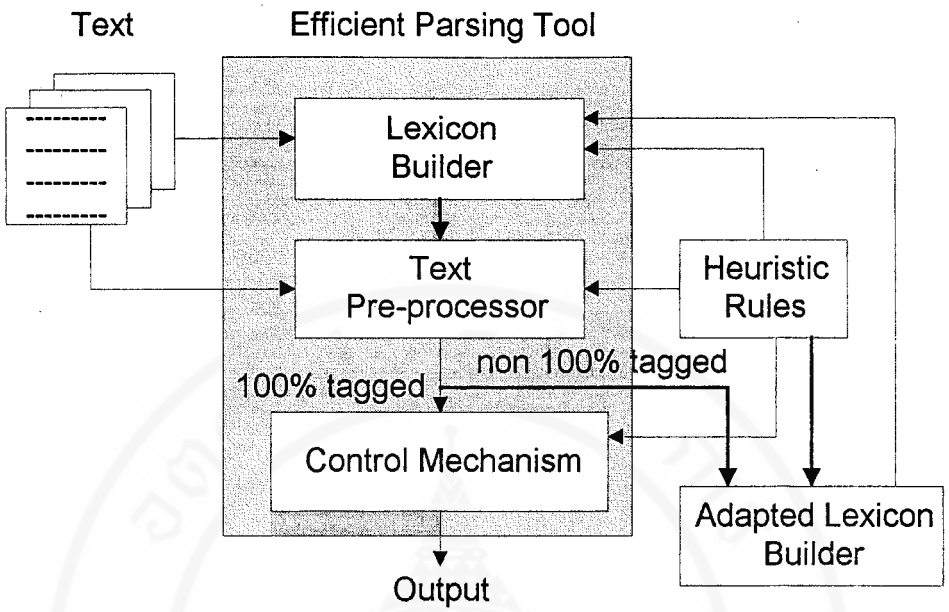

Figure 4.1 demonstrates the overall system of the Text Efficient Parsing Tool.

The system consists of five major components. They are as follow:

1) Lexicon Builder
2) Text pre-processor
3) Control mechanism
4) Heuristic Rules
5) Adapted Lexicon Builder

1) Lexicon Builder

Lexicon builder demonstrates the ideas in building automated learnable lexicon using heuristic rules for text parsing resource. The rules are divided into three categories, based on the syntactic class of word. They are noun, verb, and adjective categories.

2) Text Pre-processor

Text pre-processor demonstrates the methods in phrases grouping to form mark-up text before parsing. It is responsible for checking the correctness of word tagging by comparing location of that word corresponding with its part-of-speech. The results are divided into two groups: 100% tagged sentences and non 100% tagged sentences.

3) Control Mechanism

For 100% tagged sentences, this mechanism controls the correctness of the parsing results by reducing the complexity of sentence and checking grammar patterns. It also deals with word disambiguation.

4) Adapted lexicon Builder

For non 100% tagged sentences, this component deals with unknown part-of-speech of word in noun phrase grouping. According to heuristic rules, its function is responsible for finding new word to add into lexicon.

5) Heuristic Rules

The efficient and effective power of TEPT parsing system depends on this component because the heuristic rules were created to use within the above four components.

We will discuss these components in detail in the next section.

4.2. Lexicon Builder

4.2.1 Overview

Lexicon contains the grammatical and semantic information about words or word strings. Knowledge about words is the most important form of knowledge for lexicon. A good lexicon in text parsing task should guide the parser through an input text for their part-of-speech. Moreover, it should determine the appropriate meanings for words and phrases in the text in which they appear.

4.2.2 The Dynamic Learnable Lexicon Features

Two key features distinguish the ideal lexicon for text parsing. First, the lexicon must include both static and dynamic components. Second, the lexicon must be self-training or self-learning to find a new word.

- Learnable Feature

Lexicons must be self-learning or can be automated because manual developed lexicon is a time-consuming process that may require several years. The design and content of lexicons are a paradigmatic problem in knowledge representation for AI

systems. A complete lexicon needs experts who are highly experienced to create and examine. It takes a long time to complete the job. A moderate-sized lexicon, with the ability to find new words, is a good choice for solving this problem.

- Dynamic Feature

Lexicons must be dynamic because the space of possible word in each domain specific is unbounded. First, it is essential to cover all of common words that might appear in an arbitrary text of each domain. These common words will be used preliminary in parsing system. Then, the lexicon will extend their capacity into domain specific keywords.

The domain specific lexicon is appropriate for dynamic feature when we consider with size of resources and time spent in acquiring. The gathering of these domain specific lexicons makes use of the overall knowledge in the world.

4.2.3 The Major and Minor Components in Lexicon

The important categories for POS in any dictionary or lexicon are nouns, verbs, pronouns, adjectives, adverbs, conjunctions, articles and prepositions. We divide these categories into two groups as described in Figure 4.2: major and minor. **Major groups** are nouns, verbs and adjectives. **Minor groups** are pronouns, conjunctions, articles, prepositions and adverbs. Major groups are the semantic component or keyspot in text. On the other hand, minor groups give the minor semantic component.

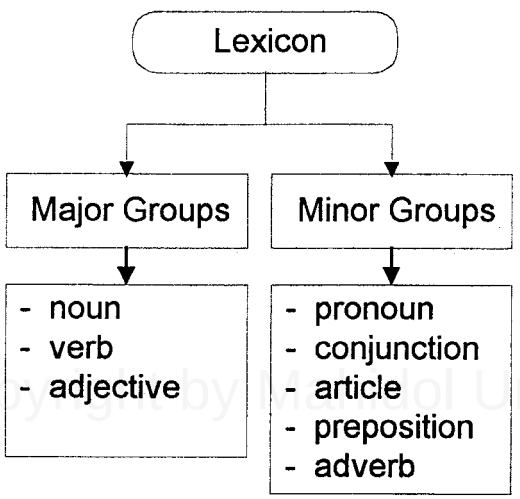

Figure 4.2 : Group of Words in Lexicon

In any domain keywords, nouns and verbs are the main idea in a sentence. Nouns tell who have done something. Verbs are the action words in a statement. They tell what is happening, what a noun is doing or what is being done to it. There are noun groups, also called noun phrases which means that two or more nouns, or a noun and an adjective, are put together to form one noun.

Adjectives are classified into major group because they are used in noun phrases. Adjectives are descriptive words, sometimes called modifiers. They add detail to noun phrases or sentences.

4.2.4 The proposed model for Lexicon Design by Using Heuristic Rules

As we described above, the most significant parts in any sentence are major groups (noun, verb, adjective, and adverb). With this idea, we will generate a dynamic and learnable lexicon of major groups by using heuristic rules. Figure 4.3 shows the lexicon building which can be divided into 4 groups: Common, Noun, Adjective, and Verb.

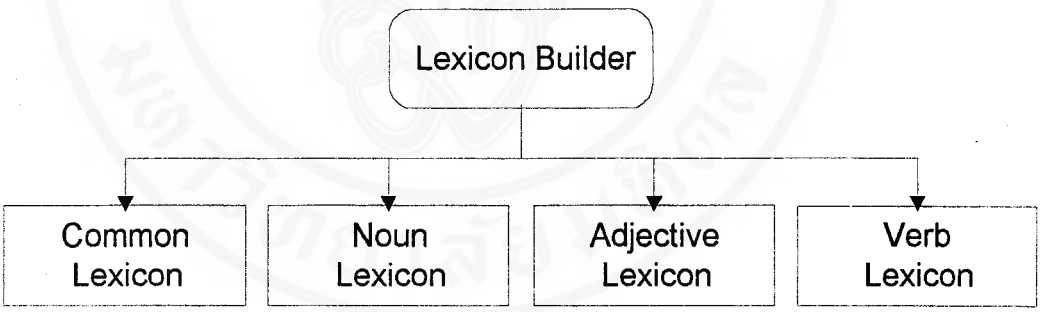

Figure 4.3 : The components of lexicon builder

4.2.4.1 Common Lexicon Builder

Common Lexicon built manually by collecting the minor groups (pronouns, conjunctions, articles and prepositions) into lexicon. Some general phrases (would like to, be able to, have to, in particular etc.) also store too. The user can add, delete or change word lexicon by himself.

4.2.4.2 Noun Lexicon Builder

We propose 21 heuristic rules for building noun lexicon as shown in Table 4.1.

Table 4.1 : Heuristic Rules for Noun Lexicon Builder

Rule No.	Pattern
1.1	ART-X-AUX >> X is noun
1.2	ART-X-OF >> X is noun
1.3	word ending with -ance
1.4	word ending with -bility
1.5	word ending with -ble
1.6	word ending with -dom
1.7	word ending with -ence
1.8	word ending with -er
1.9	word ending with -hood
1.10	word ending with -ism
1.11	word ending with -logy
1.12	word ending with -ment
1.13	word ending with -ness
1.14	word ending with -or
1.15	word ending with -ship
1.16	word ending with -sion
1.17	word ending with -ter
1.18	word ending with -tion
1.19	word ending with -tor
1.20	word ending with -ture
1.21	word ending with -ty

where ART, X, AUX, OF refer to article, unknown word, auxiliary verb, and 'of' respectively in any sentences. By scanning the arbitrary texts, we look for these heuristic patterns. Each time a pattern is found, a word is extracted from the text and examined with a standard dictionary before added to a noun lexicon. Then we compute the correct statistics of each word to evaluate each heuristic rule.

4.2.4.3 Adjective Lexicon Builder

We propose 27 heuristic rules for building adjective lexicon as depicted in Table 4.2.

Table 4.2 : Heuristic Rules for Adjective Lexicon Builder

Rule No.	Pattern
2.1	AUX-X-TO >> X is adjective
2.2	ART-X-ADJ >> X is adjective
2.3	IT-IS-X >> X is adjective

Table 4.2 : Heuristic Rules for Adjective Lexicon Builder (Continue)

2.4	word ending with -able
2.5	word ending with -al
2.6	word ending with -ant
2.7	word ending with -ative
2.8	word ending with -cal
2.9	word ending with -ent
2.10	word ending with -ful
2.11	word ending with -ial
2.12	word ending with -ible
2.13	word ending with -ic
2.14	word ending with -ive
2.15	word ending with -less
2.16	word ending with -like
2.17	word ending with -nal
2.18	word ending with -ous
2.19	word ending with -ral
2.20	word ending with -tic
2.21	word ending with -tive
2.22	word ending with -ual
2.23	word ending with -ular
2.24	word ending with -ward
2.25	word ending with -wise
2.26	word format ??-??ed
2.27	word format ??-??ing

where ART, X, ADJ, AUX, TO refer to article, unknown word, adjective, auxiliary verb, and 'to' respectively in any sentences. By scanning the arbitrary texts, we look for these heuristic patterns. Each time a pattern is found, a word is extracted from the text and check that it is a verb or not. If it is not a verb then added to an adjective lexicon. Then we compute the correct statistics of each word to evaluate each heuristic rule.

4.2.4.4 Verb Lexicon Builder

Figure 4.4 depicts four steps in the verb lexicon builder: select verb pattern, change to candidate verb, select words for testing, and test true verb steps.

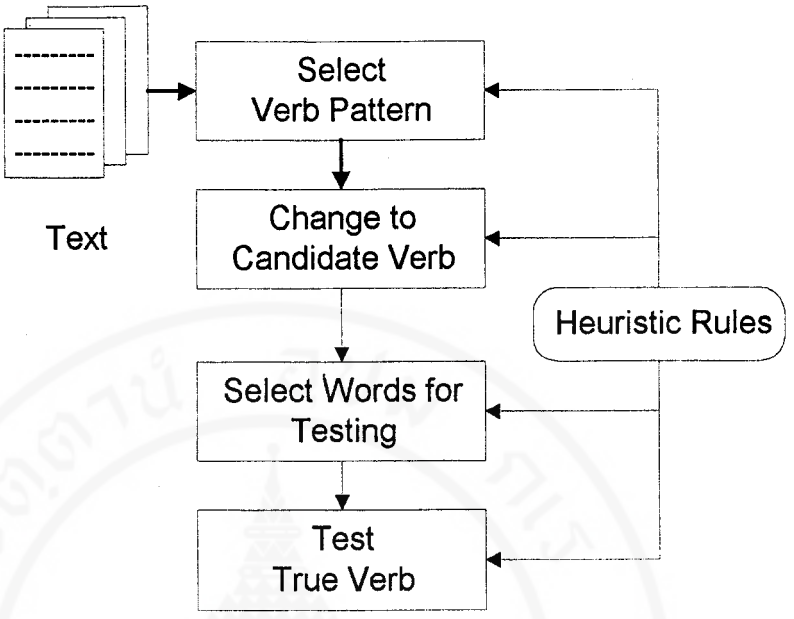

Figure 4.4 depicts the model for learnable verb lexicon design

4.2.4.4.1 Select Verb Pattern

There are many verb patterns in English grammar as shown in Table 4.3 and Table 4.4 below.

Table 4.3 shows verb patterns in different tenses

	Present	Past	Future
Simple	V1	V2	will + V1
Continuous	is + V(ing)	was + V(ing)	will be + V(ing)
Perfect	has + V3	had + V3	will have + V3
Perfect Cont.	has been + V(ing)	had been + V(ing)	will have been + V(ing)

Table 4.4 shows passive voice verb patterns in different tenses

	Present	Past	Future
Simple	is + V3	was + V3	will be + V3
Continuous	is being + V3	was being + V3	will be being + V3
Perfect	has been + V3	had been + V3	will have been + V3
Perfect Cont.	has been being + V3	had been being + V3	will have been being + V3

Some pattern has a uniform structure and easy to identify. For example, present continuous tense begins with auxiliary verb and ends with "ing" ending verb (or gerund). So,

If we get a continuous present tense verb pattern from any text then we can assume that the last position word of that pattern with "ing" ending is a verb.

With this approach, we can find another verb by using a verb form in the table above by defining heuristic rule as follow.

Rule 3.1: The pattern, starting with keywords (such as, "is, am, are, was, were, has, have, had") and ending with "ed-ending or ing-ending" words, is verb pattern.

Rule 3.2: The last "ed-ending or ing-ending" words in pattern is verb.

4.2.4.4.2 Change to Candidate Verb

This stage is to transform candidate verb from extracted verb patterns. When we get verb patterns from any sentence, most of these verb patterns end with "ed" or "ing". We transform extracted verbs into candidate verbs with the following heuristic rules:

Rule 4.1: If the extracted verb ends with "ed" then the candidate verb is transformed into two forms: "ed-removed" and "ed-removed with plus e".

For example, the extracted verb "trained" is transformed into "train" and "traine".

Rule 4.2: If the extracted verb ends with "ied" then the candidate verb is transformed into two forms: "y" and "ye".

For example, the extracted verb "dried" is transformed into "dry" and "drye"

Rule 4.3: If the extracted verb ends with "ing" then the candidate verb is transformed into two forms: "ing-removed" and "ing-removed with plus e".

For example, the extracted verb "training" is transformed into "train" and "traine".

Rule 4.4: After performing "ed-removed" and "ing-removed", if the transformed verb ends with these characters, such as "dd", "gg", "ll",

"mm", "nn", "pp", "rr", "ss", and "tt", then the candidate verb is transformed into two additional forms: "last remove with plus e", and "last remove".

For example, the transformed verb ends with "tt" such as "cutt". It is tranformed into four forms: "cutte", "cutt", "cut" and "cute".

Table 4.5 summarizes the heuristic rules to change candidate verb form

Ending Characters	Candidate Verb
-ed	ed-removed ed-removed with plus "e"
-ied	change to "y" change to "ye"
-ing	ing-removed ing-removed with plus "e"
-dded, -dding (include -gg, -ll, -mm, -nn, -pp, -rr, -ss, -tt)	ed-removed, ing-removed plus "e" last character removed last character removed with plus "e"

4.2.4.4.3 Test True Verb

In this stage, we will illustrate the learning process for finding true verb from candidate verb. We propose two heuristic rules to select testing words: word follow "to" and word ending with "s" because these words have tendency to be true verb.

- Word Ending With "s"

We look for any "s" ending words in any sentence, and test them with words transformed from section 4.1.4.2. By changing candidate words into singular verb from the following rules:

Rule 5.1: If verb ends with "e" then changes to "es"
Rule 5.2: If verb ends with "y" then changes to "ies"
Rule 5.3: If verb ends with "ss" then changes to "sses"
Rule 5.4: If verb ends with "sh" then changes to "shes"
Rule 5.5: If verb ends with "ch" then changes to "ches"
Rule 5.6: If verb ends with "x" then changes to "xes"

Rule 5.7: If verb ends with characters other than the above then add "s" (General case)

Finally, we compare the processed verb with the candidate verb. If they match then this candidate verb is a true verb and stored in the verb lexicon.

- Word Follow "TO"

In the case of word follow "to", we compare with candidate verb. If they match then add that candidate verb to the verb lexicon as another true verb.

Rule 6.1: TO-X >> X is verb

where X refer to unknown word.

4.3 Text pre-processor methods

Text is produced through uniform structure, starting at the story level or paragraph level and working down to the sentence level or the level of individual words and phrases. With the fact that people read text from left to right, skim for particular pieces of information, look back for other parts, and look ahead to get the important parts in the text body. The correct interpretation in skimming of the text depends on several factors such as length, style, complexity of sentences, and the flow between paragraphs.

In other cases, pre-processor process reduces the complexity of parsing and helps produce a reasonable interpretation. Pre-processor process may not be necessary if parsing was not so hard. But in the real world, this task does not occur frequently. We start the input text from single-pass left-to-right strategies, making multiple passes through bodies of text, and using coarse features of the text such as phrases and common words to help guide correct interpretation. Two aspects of pre-processor seem to have particular promise for the quality and efficiency of later processing. They are word tagging and phrase grouping.

As shown in Figure 4.5, the text pre-processor method consists of three major modules: word tagging, location checking and phrase grouping.

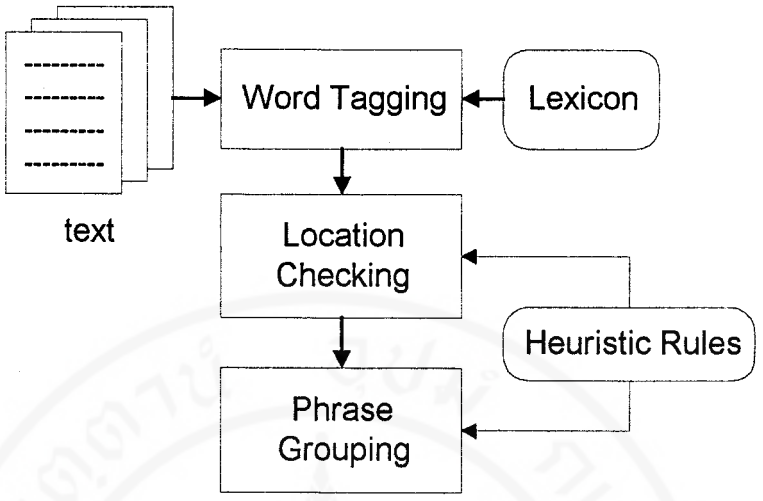

Figure 4.5 illustrates the text pre-processor methods

4.3.1 Word Tagging

In the word-tagging module, it reads the sentence from the text and assigns syntactic categories to each word in the sentence by looking up in the lexicon (common, noun, verb, and adjective) which constructed by the above section.

4.3.2 Location Checking

To examine the correctness of word tagging, we check the word again by using the location checking to resolve word ambiguities. This method is an attempt to improve the tagging accuracy. The heuristic rules for location checking are shown in Table 4.6 below:

Table 4.6: Heuristic rules for location checking

Rule No.	Pattern
7.1	<DET/VERB> verb after determiner is noun
7.2	<DET/END-VERB: ed or ing> verb after determiner is adjective
7.3	<"of"/VERB: V3> V3 after "of" is adjective
7.4	<ADJ/VERB> verb after adjective is noun
7.5	<ADJ/END-VERB: ed> verb-ed after adjective is adjective
7.6	<PP/VERB> verb after preposition is noun
7.7	<VERB/"of"> verb before "of" is noun

Table 4.6: Heuristic rules for location checking (Continue)

7.8	<VERB/AUX> verb before auxiliary is noun
7.9	<VERB/VERB: V3> verb before V3 is noun
7.10	<Capital Letter> word begin with capital letter is noun
7.11	<ADJ/AUX> adjective before auxiliary is noun
7.12	<these/CONJ> "these" before conjunction is noun
7.13	<'S> word end with 's is adjective
7.14	<DET/PP> preposition after determiner is adjective
7.15	<AUX/AUX> auxiliary after auxiliary is verb
7.16	<TO/AUX> auxiliary after to is verb
7.17	<of/CONJ> conjuction after "of" is adjective
7.18	<VERB: V1/VERB: V1> V1 after V1 is noun
7.19	<"of"/VERB> verb after "of" is noun
7.20	<AUX/VERB: Vs> verbs after auxiliary is noun
7.21	<ADJ/and/ADJ> and between adjective is adjective
7.22	<ADJ/,/ADJ> comma between adjective is adjective
7.23	<ADJ/or/ADJ> or between adjective is adjective
7.24	<ADJ/of> adjective before "of" is noun
7.25	<CONJ/of> conjuction before "of" is noun
7.26	<DET/VERB: V3> V3 after determiner is adjective
7.27	<of/VERB: Ved> verb-ed after "of" is adjective
7.28	<NUMBER> number is noun

4.3.3 Phrase Grouping

Phrase grouping groups the text around words that are the same units. It combines words into noun phrases, verb phrases and other important phrases. This grouping helps reducing unnecessary parsing, and focusing on important phrases during parsing. The ultimate goal of phrase grouping is to effectively reduce the number of parsing decisions that must be made and to guide those decisions.

Our system uses a pattern-matching algorithm for identifying grammatical constructions especially noun phrases in text. This is because these noun phrases often play an important role as key information in text.

The following is a sample text with the results of the tagging and phrase generation of pre-processing

Original text:

A significant limitation of neural networks is that the representations are usually incomprehensible to humans. We present a novel algorithm, Trepan, for extracting comprehensible representations from trained neural networks. Our algorithm uses queries to induce a decision tree that approximates the concept represented by a given network. Our experiments demonstrate that Trepan is able to produce decision trees that maintain a high level of fidelity to their respective networks. Unlike previous work in this area, our algorithm is general in its applicability and scales to large networks and problems with high-dimensional input spaces.

Pre-processing text:

[A significant limitation] of [neural networks] is that [the representations] are usually incomprehensible to [humans]. We present [a novel algorithm], [Trepan], for extracting [comprehensible representations] from [trained neural networks]. [Our algorithm] uses [queries] to induce [a decision tree] that approximates [the concept] represented by [a given network]. [Our experiments] demonstrate that [Trepan] is able to produce [decision trees] that maintain [a high level] of [fidelity] to [their respective networks]. Unlike [previous work] in [this area], [our algorithm] is general in [its applicability] and scales to [large networks] and [problems] with [high-dimensional input spaces].

The noun phrases may be combined from the following heuristic patterns:

Rule 8.1: Noun Phrase -> DET | ADV | ADJ | Noun
Rule 8.2: Noun Phrase -> Noun | Noun

To combine noun phrases in the sentence, the system uses word delimeters to define phrase boundaries in the text. Determiners (e.g., "the" , "an", and "a"), preposition, conjunctions (e.g., "and" and "or"), and certain punctuation (e.g., colon, semicolon, or parenthesis) help to determine these boundaries by marking the beginning and ending of a phrase.

The verb phrases may be combined from the following heuristic patterns:

Rule 9.1: Verb Phrase -> AUX | ADV | <END-VERB: ed or ing>
Rule 9.2: Verb Phrase -> AUX | ADV | <VERB: V2 or V3>

4.4 Control Mechanism

As shown in Figure 4.6, we propose four approaches to control the correctness of text parsing: noun-verb disambiguation, sentence decomposition, preposition-noun phrase removal and grammar checking.

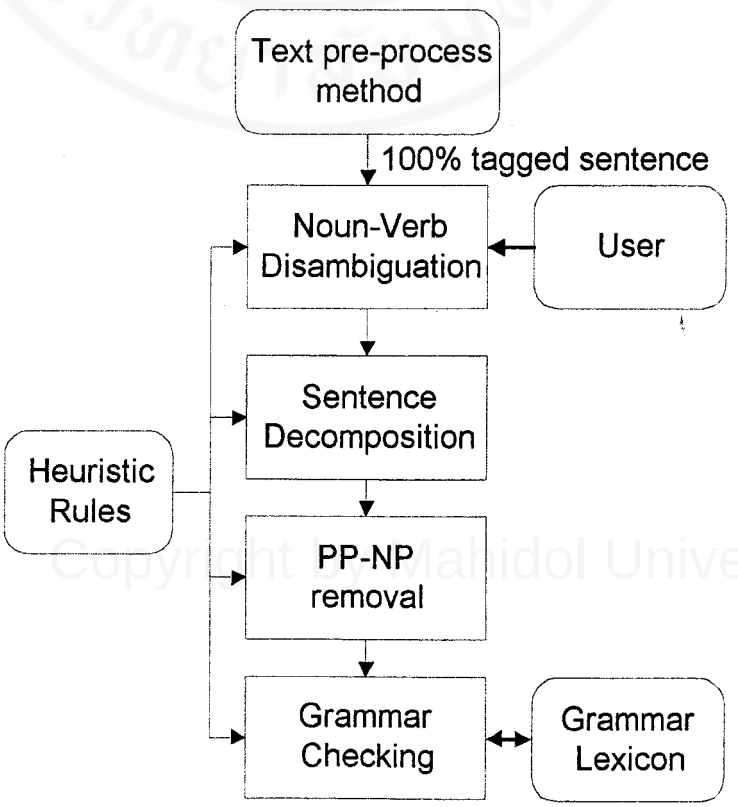

Figure 4.6 shows the control mechanism for text parsing

4.4.1 Noun-Verb Disambiguation

Because some words are ambiguous, and this ambiguity can cause the incorrect parsing. Verbs, such as control, design, result etc., can function as noun. To reduce this effect, we use supervised learning to judge for the correct sense of the ambiguous word. As mentioned above, we are interested in noun and verb because they are the most important components in sentence. Thus, we will mainly emphasize on verb ambiguous word.

4.4.1.1 Check Noun-Verb Disambiguation by the User

After performing text pre-processing, the system produces a syntactic analysis for the sentence and outputs it in the grammar pattern. When the user has found the ambiguous verb that obtained from incorrect parsing, he can mark it with ambiguous flag. The parsing system will adjust that ambiguous verb to noun, and perform parsing process again.

4.4.1.2 Adjust for Noun-Verb Disambiguation

After found ambiguous verb that was checked by the user, the system changes it to noun and combines it with pre- or post-position nouns to form new noun phrase.

Here are the combine patterns for noun-verb disambiguation:

Table 4.7: Heuristic rules to combine patterns for noun-verb disambiguation

Rule No.	Pattern
10.1	[Noun] + [N-V verb] (combine with pre-position)
10.2	[N-V verb] + [Noun] (combine with post-position)
10.3	[Noun] + [N-V verb] + [Noun] (combine with pre- and post- position)
10.4	[N-V verb] (change to noun)

4.4.2 Sentence Decomposition

In general, any sentence can be classified into three structures: simple, compound, and complex. A simple sentence contains only a single clause, while a compound sentence and a complex sentence

contain at least two clauses joined by conjunctions such as "and, but, and or". The grammar pattern of simple sentence is easily classified because it contains only one noun and one verb. Compound sentence is also easy because we can separate it into two simple sentences by removing conjunction. But complex sentence is difficult to classify.

The objective in sentence decomposition is to transform the compound and complex sentence into many simple sentences. With this criteria, the parsing process is easy to understand and check for the correctness of parsing sentence. The heuristic rules for sentence decomposition are illustrated in Table 4.8 below.

Table 4.8: Heuristic rules for sentence decomposition

Rule No.	Pattern
11.1	Conj-Noun
11.2	Conj-Pronoun
11.3	Conj-VP
11.4	Conj-V1
11.5	Conj-Vs
11.6	Conj-Aux
11.7	Conj-Ving
11.8	Conj-Ved
11.9	Noun-V3
11.10	Noun-Ved
11.11	Noun-Ving
11.12	Noun-Aux
11.13	Noun-VP
11.14	Noun-Noun
11.15	Noun-Pro

Let us consider the following examples:

"The checksum is recomputed by the receiving hardware and is checked against the transmitted checksum."

After performing text pre-processing, the system produces a syntactic analysis for this sentence and output it in the pattern as follow:

[The checksum] [is recomputed] by [the receiving hardware] *and* [is checked] against [the transmitted checksum].

This sentence is decomposed into two sub-sentences by matching the rule "Conj-VP" as follow:

"The checksum is recomputed by the receiving hardware."
"<NP> is checked against the transmitted checksum."

4.4.3 Preposition-Noun Phrase Removal

To reduce the complexity and size of pattern, we remove the PP-NP from the sentence.

Let us consider the following examples:

"Algorithm C was introduced by the authors in 1983."

After performing text pre-processing, the system produces a syntactic analysis for this sentence and output it in the pattern as follow:

[Algorithm C] [was introduced] by [the authors] in [1983]
NP VP PP NP PP NP

After pp-np removing, we output the new results as:

[Algorithm C] [was introduced]
NP VP

This sentence is represented with grammar pattern <NP VP>. The user examines whether a pattern should be accepted or rejected. If the user accepts this pattern as correct then the system stores them in the grammar lexicon.

Table 4.9 depicts the heuristic rules for preposition-noun phrase removal.

Table 4.9: Heuristic rules for preposition-noun phrase removal

Rule No.	Pattern
12.1	[PP-NP]
12.2	[PP-NP]-PP
12.3	[PP-NP]-To

With this approach, we can store any correct grammar pattern in the database called grammar lexicon. Using these grammar patterns we can match for the unknown pattern in any text.

4.4.4 Grammar Checking

By comparing the unknown pattern with the grammar lexicon created from the above section. If it exists in the lexicon, we expect that

the unknown pattern is correct and parsing is complete. We select noun phrases in sentence as significant parts. These extracted noun phrases can be stores or used in domain specific lexicon.

4.5 Adapted Lexicon Builder

At the completion of text pre-processor, each sentence is divided into two sets of sentences: 100% tagged sentences and non 100% tagged sentences. The 100% tagged sentences were processed by control mechanism routine for final outputs but the non 100% tagged sentences were processed by this routine. The goal of this routine is to learn new word lexicon by using heuristic rules. Like lexicon builder routine, adapted lexicon builder searches for unknown word matched with heuristic rules shown in Table 4.10 in sentence and uses heuristic rules from Table 4.11 to classify unknown words as noun or Table 4.12 to classify unknown words as adjective.

Table 4.10: Heuristic rules to select unknown words

Rule No.	Pattern
13.1	[ADJ-X]
13.2	[NP-X]
13.3	[X-NP]
13.4	[ART-X]

Table 4.11: Heuristic rules to classify unknown words as Noun

Rule No.	Pattern
14.1	[ADJ-X] [NP-X]
14.2	[NP-X] [X-NP]
14.3	[ART-X] [NP-X]

Table 4.12: Heuristic rules to classify unknown words as Adjective

Rule No.	Pattern
15.1	[X-ADJ] [X-NP]

CHAPTER V

SYSTEM DESIGN

In this chapter, we describe how to design the system according to the TEPT architecture mentioned in the previous chapter. We will present the routines for lexicon acquisition and parsing system and show procedures of how the texts are processed or examined.

5.1 System Structure

The context diagram of proposed TEPT parsing system is shown in Figure 5.1. The system consists of two parts: a learning phase and a testing phase. In learning phase, there is one routine: lexicon builder. In testing phase, it consists of two routines: text pre-processor, and control mechanism.

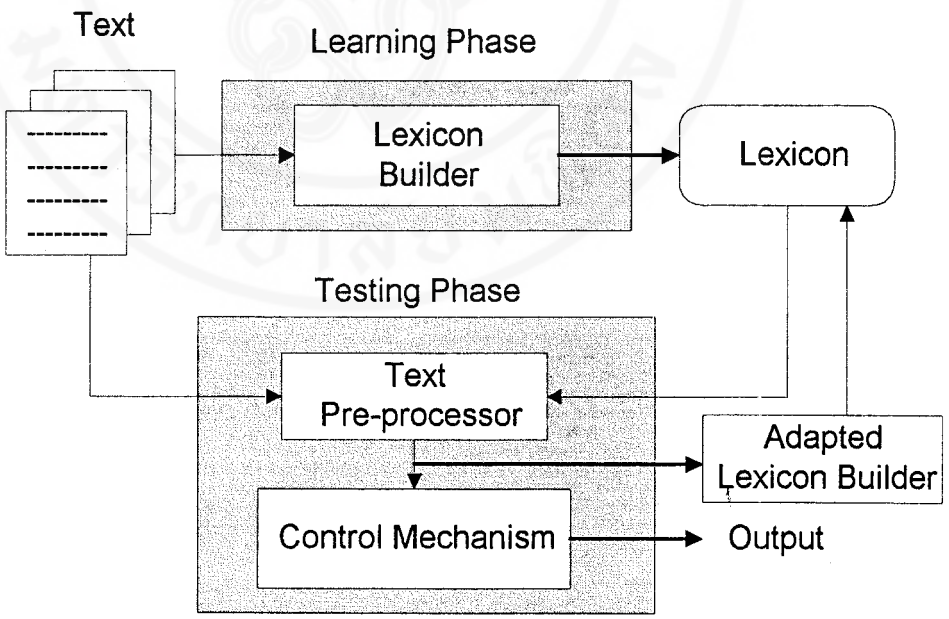

Figure 5.1 illustrates the context diagram of TEPT parsing system.

The structure chart of the parsing system is demonstrated in Figure 5.2. There are four main routines: lexicon builder, text pre-processor, control mechanism and adapted lexicon builder. The lexicon builder routine consists of five modules: common lexicon, noun lexicon, adjective lexicon, verb lexicon, and grammar lexicon. The text pre-processor routine also consists of three modules: word tagging, location checking, and phrase grouping. The last

routine, control mechanism, consists of four modules: noun-verb disambiguation, sentence decomposition, PP-NP removal and grammar checking. These modules will be explained in the next section.

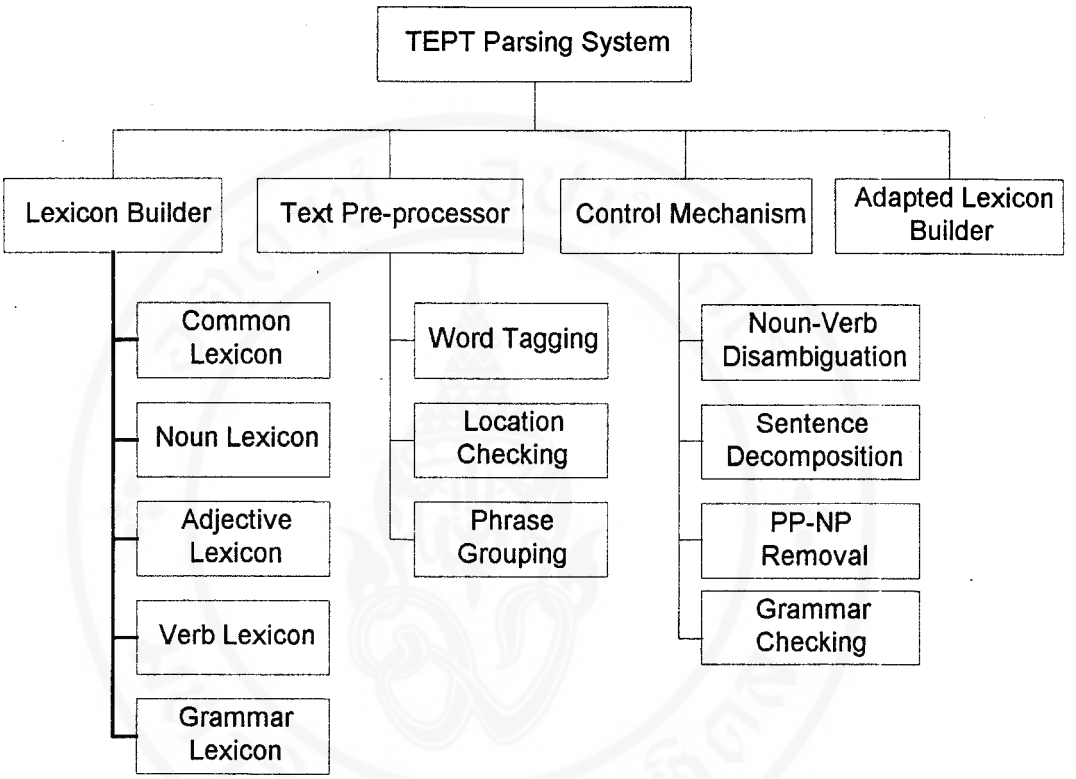

Figure 5.2 : structure chart of the TEPT parsing system

Dependency Diagram of heuristic rules used in TEPT is illustrated in Figure 5.3. This diagram demonstrates the relationship of heuristic rules in each module and path between each sub-module in TEPT operation. The overall heuristic rules used in TEPT are 124 rules. The rules are divided into 4 groups: 62 rules for lexicon builder, 32 rules for text pre-processor, 22 rules for control mechanism, and 8 rules for adapted lexicon builder.

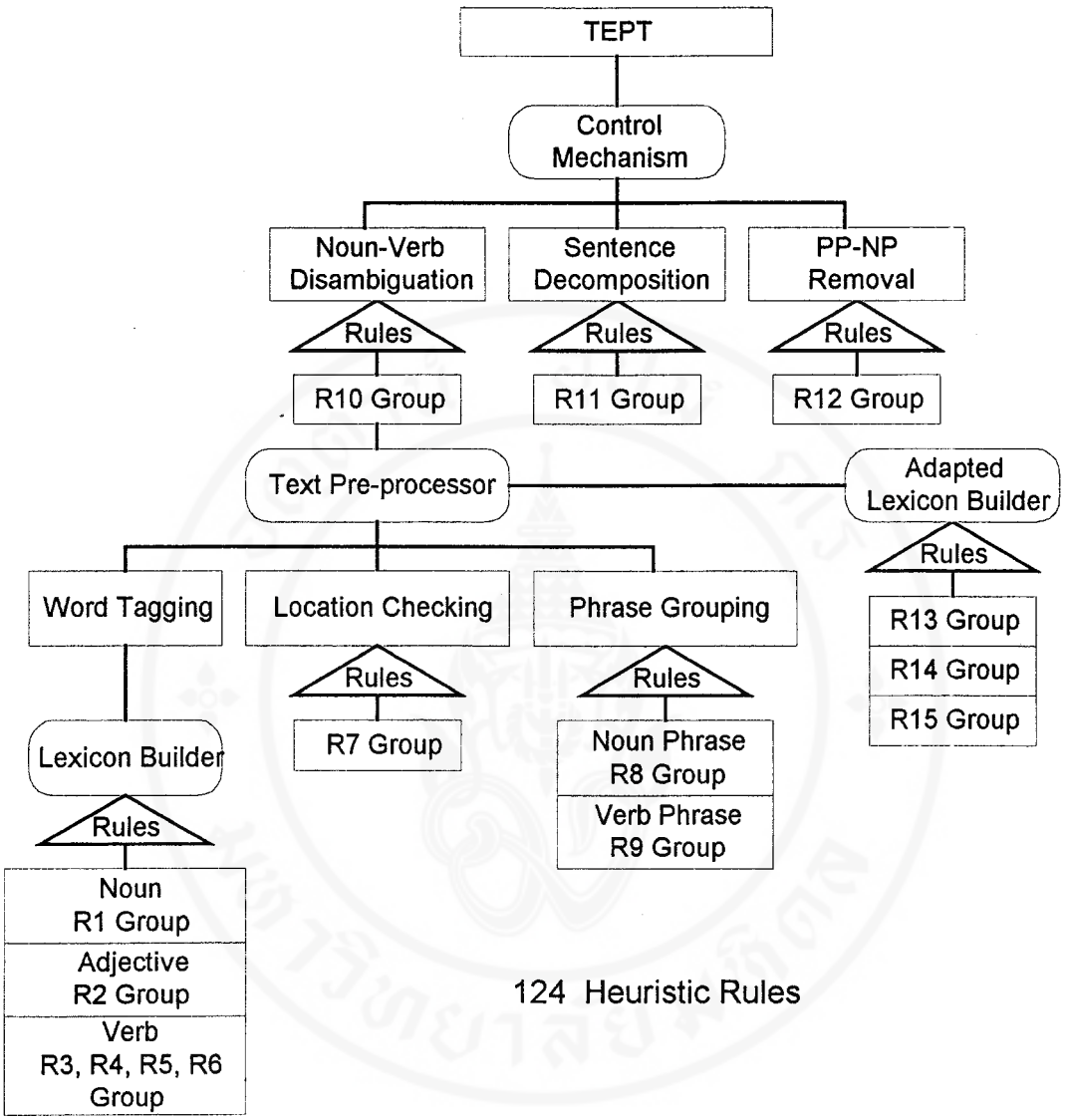

Figure 5.3: Dependency Diagram of heuristic rules used in TEPT

5.2 Learning Phase

The learning phase is responsible for domain lexicon acquisition from a set of training domain texts (Computer Science domain). This phase learns new domain words by using the heuristic rules in lexicon builder routine and uses machine learning technique to get from training data set. The design of learning phase is illustrated in Figure 5.4.

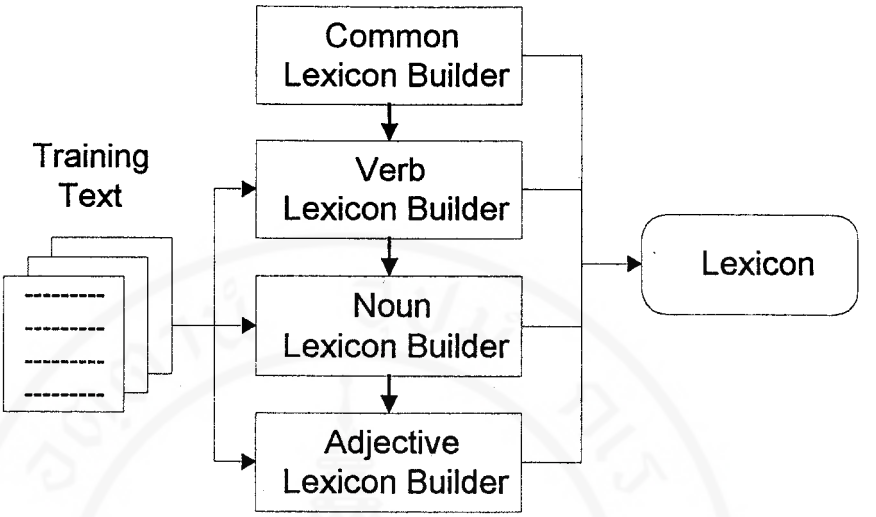

Figure 5.4: Design of Learning Phase

Learning phase routine consists of four modules: common lexicon builder, noun lexicon builder, adjective lexicon builder, and verb lexicon builder.

In common lexicon builder, we fill our lexicon with word in minor groups such as pronouns, conjunctions, adverbs, articles and prepositions. In addition, we also collect some general phrases because much of the common phrases in any text cause an incorrect result if we tag them in each one word in text parsing.

At the beginning, we start with automated learnable verb lexicon acquisition by searching verb pattern from a set of domain training texts with the criteria according to heuristic rules (R3 group) and transform the last position word of verb pattern into candidate verb by using heuristic rules (R4 group). After that, we select testing words which have high tendency to be true verb from training texts and compare them against candidate verbs to find true verb. The process, for finding true verb from candidate verb in order to add them in verb lexicon, based on heuristic rules (R5 and R6 group).

The reason in building verb lexicon is to use them in identifying noun phrases boundary in sentence and to reduce noun-verb ambiguation. Moreover, they are used to develop grammar patterns.

After we have the suitable number of verb lexicon, noun and adjective lexicon will be developed later. By comparing any sentences in a set of training texts with heuristic rules (R1 group for noun, R2 group for adjective), we can find new learning words to add to the lexicon. For each new extracted word, we

confirm the correct part-of-speech from a standard dictionary. In case of new noun which not found in standard dictionary, we confirm from domain specialists.

As a result, the lexicon will be promptly used in the next phase, testing phase, to test TEPT's performance.

5.3 Testing Phase

In testing phase, we select about 1500 sentences from domain training texts to test the performance of TEPT. The design of testing phase is presented in Figure 5.5.

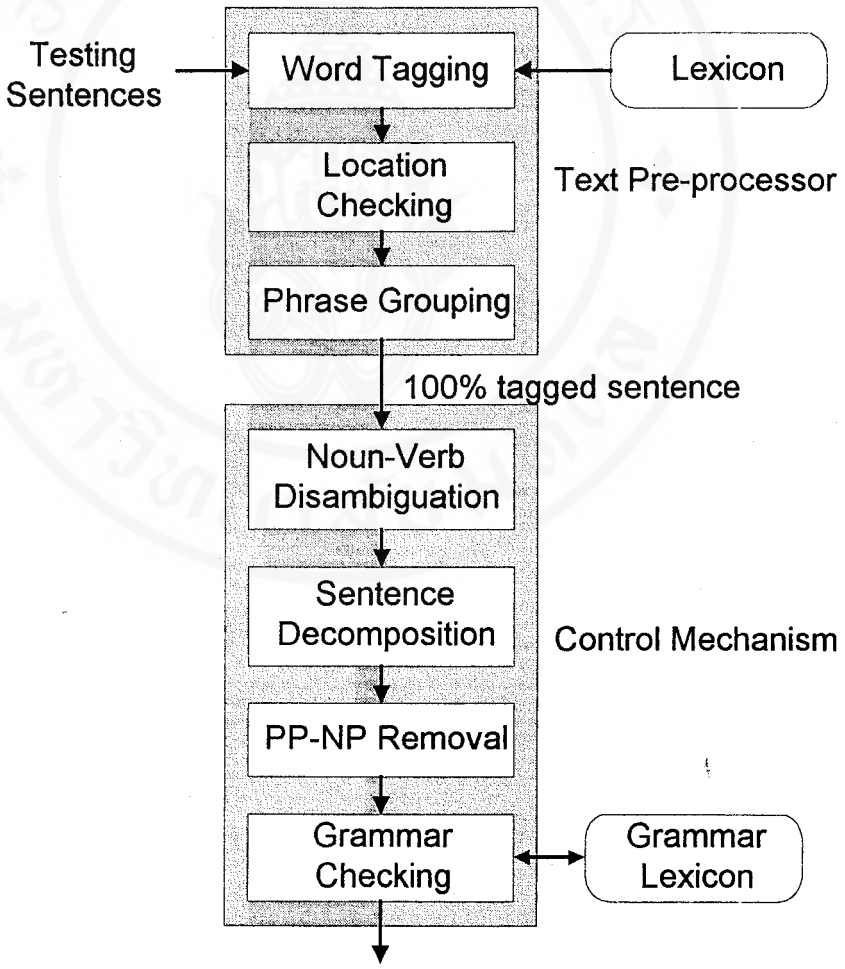

Figure 5.5: Design of Testing Phase

This phase consists of two routines: text pre-processor and control mechanism. The text pre-processor routine consists of three modules: word tagging, location checking, and phrase grouping. The control mechanism routine

consists of four modules: noun-verb disambiguation, sentence decomposition, PP-NP removal and grammar checking.

Text pre-processor routine uses domain lexicon resources from the above section to recognize prepositions, common phrases, noun groups, and verb groups in sentences before parsing.

Word tagging module scans for word delimeters and common phrases in testing sentence. Word delimeters consist of white space symbol, i.e. space, tab, or newline characters. Punctuation such as full stop, parentheses, question mark, etc., are also word delimeters. Next, TEPT assigns syntactic categories to each word in the testing sentence by looking up in the lexicon (common, noun, verb, and adjective).

Location checking module checks the correctness of word tagging by using location checking heuristic rules (R7 group) for resolving word ambiguities. It changes part-of-speech of word that matches with heuristic rules to new part-of-speech. After that, the result is transferred to the next module.

Phrase grouping module combines the modifiers in the sentence i.e. article, adjective, and adverb, with the head noun to form noun phrases according to heuristic rules (R8 group). This module also combines verb phrases according to heuristic rules (R9 group).

At the end of text pre-processor routine, the results were divided into two sets of sentences: 100% tagged sentences and non 100% tagged sentences. The control mechanism routine selects only 100% tagged sentences to process.

Control mechanism routine helps TEPT in control the correctness of parsing results which are represented to the user in grammar pattern forms. If error results are produced at this stage, the user should do the necessary adjustments to the results until it produces the output correctly or discards that error result.

The noun-verb disambiguation module allows the user to judge for the correct semantic sense of the ambiguous verb. The user can mark flag for that ambiguous verb by himself and the system will combine it with pre- or post-position nouns to form new noun phrases according to heuristic rules (R10 group).

The sentence decomposition module transforms the compound and complex sentence into many simple sentences by using pre-defined heuristic rules (R11 group). It helps the system in reducing the complexity of the sentence.

The PP-NP removal module removes preposition-noun phrase pattern from the sentence in order to reduce the size of sentence and minimize unnecessary parsing. It helps the system in checking for the correctness of the grammar pattern parsing result. The removal criteria are defined in heuristic rules (R12 group). After performing this module, the parsing outputs are represented in grammar patterns.

TEPT checks grammar patterns in output sentence by comparing with grammar lexicon. The grammar lexicon is used in control mechanism routine for checking correctness of the parsing result. This grammar lexicon stores non-complex grammar patterns. We select them from final parsing result and add to the lexicon. For example, if TEPT founds unknown grammar patterns in parsing output, TEPT will report unknown grammar pattern to the user. The user checks for correctness, if that unknown grammar pattern is correct, then the user adds to grammar lexicon, otherwise, discards it. This method, called supervised learning by the users, is used in grammar lexicon builder.

The noun phrases in final parsing sentence with correct grammar pattern are extracted from the result and examine by user before add into domain specific lexicon.

5.4 Adapted Lexicon Builder

After performing text pre-processor, the results were divided into two sets of sentences: 100% tagged sentences and non 100% tagged sentences. The Adapted lexicon builder routine selects only non 100% tagged sentences to process. By searching for unknown word in noun phrases according to heuristic rules criteria (R13 group), adapted lexicon builder uses heuristic rules (R14 group) to classify these unknown words as noun and heuristic rules (R15 group) to classify as adjective. The new noun or adjective words are stored in lexicon.

5.5 The Dynamic Characteristic in TEPT

As described above, TEPT acquires new knowledge lexicon by using heuristic rules. The other characteristic of TEPT is dynamic feature. As depicts in figure 5.6 below, TEPT generated domain lexicon for each domain specific text based on training texts in learning phase.

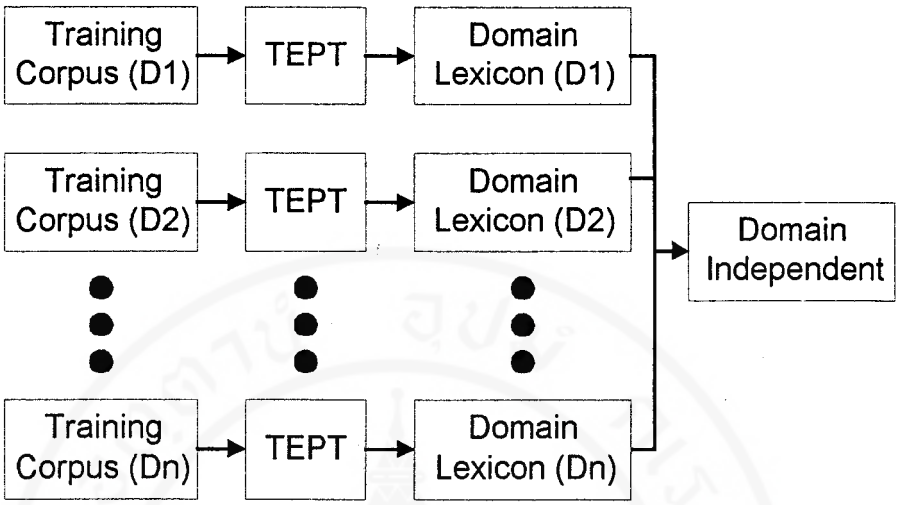

Figure 5.6: Dynamic Lexicon

The number of domain specific to add in domain lexicon is based on users' consideration and size of lexicon. Each user can create his/her domain specific lexicon depending on their interested topic. At different time and different set of training texts, the content of lexicon will be changed. This called dynamic property of lexicon.

The complete dynamic lexicons, include n-th domain specific, will cover all domain specific lexicons. These lexicons will be called domain independent lexicon because they gather all knowledge over the text world. "Everything must be found in here" is correctly explained for independent lexicon concept.

The individual domain lexicon is suitable for tasks in information management because it can be used in IR, IE and TC. IR and TC search for keywords which represent the significant contents in texts to classify the relevant text. IE extracts information from text by using extraction patterns based on keywords. These keywords are domain specific lexicon. So, domain specific lexicon will be used as basis tool for information management.

CHAPTER VI

EXPERIMENTAL RESULT

6.1 Objectives

The objectives of the experiments are as follow:

- To demonstrate that the concept of automated resource acquisition in TEPT is appropriate to the natural language task.
- To measure how correctly the heuristic rules, which used in parsing system, perform.
- To measure the parsing efficiency in extracting noun phrases from the sentence.
- To demonstrate that extracting noun phrases is appropriate to the domain specific tasks.

6.2 Method

Text documents are essentially required in the experiment. Approximately 800 text documents from Computer Science journals are used to test on the system. The experiment is divided into two stages: training and testing.

In training stage, a set of lexicon based on training documents is generated. The system performed automated lexicon knowledge acquisition by searching for any word that matched with pre-defined heuristic rules for each type of lexicon (noun, adjective, verb), and checked the correctness of part-of-speech with standard dictionary before added in lexicon.

In testing stage, we used the same documents of the training set. This set of data was used to generate lexicon to test performance of text parsing as well. First, we processed each text with word tagging from our lexicon, and then performed with word location checking and phrase grouping. In this step, we called text-preprocessing step. Finally, we used the control mechanism: noun-verb disambiguation, sentence decomposition, preposition-noun phrase removal, and grammar checking, to process text and check for correctness. The output results were presented in part of noun phrase extracted and evaluated for their domain specific.

6.3 Evaluation

We collected statistics of output results by classifying each extracted item into one of two categories: correct or incorrect.

6.4 Result

The experimental result can be summarized into three main topics: lexicon builder, text pre-processor, and control mechanism.

6.4.1 Lexicon Builder

In lexicon builder, we used the training set of arbitrary texts to generate the lexicon. New extracted words were classified into each type of lexicon such as adjective, verb, and noun. The duplicated words were eliminated. We evaluated the correctness of lexicon builder by consulting a standard dictionary.

- Noun Lexicon

In noun lexicon training, the system used 21 heuristic rules to search for new nouns in training sets. Words, such as adjective, verb, or duplicated noun, were discarded. Some words, such as adverb or conjunction, were checked as incorrect noun. The remaining were added in noun lexicon and computed statistics of each rule. The results were shown in table 6.1 below.

Table 6.1 : The result of noun lexicon builder

No	Rules	Correct	Incorrect	% Correct
1.1	ART-X-AUX	200	0	100
1.2	ART-X-OF	533	0	100
1.3	ance-NOUN	133	0	100
1.4	bility-NOUN	90	0	100
1.5	ble-NOUN	26	0	100
1.6	dom-NOUN	6	0	100
1.7	ence-NOUN	168	1	99.41
1.8	er-NOUN	1126	14	98.77
1.9	hood-NOUN	20	0	100
1.10	ism-NOUN	167	0	100
1.11	logy-NOUN	71	0	100
1.12	ment-NOUN	238	0	100
1.13	ness-NOUN	417	0	100
1.14	or-NOUN	116	2	98.31
1.15	ship-NOUN	50	1	98.04

Table 6.1 : The result of noun lexicon builder (Continue)

1.16	sion-NOUN	147	0	100
1.17	ter-NOUN	275	1	99.64
1.18	tion-NOUN	1040	0	100
1.19	tor-NOUN	199	0	100
1.20	ture-NOUN	74	0	100
1.21	ty-NOUN	436	1	99.77
	TOTAL	5532	20	99.64

In our experiment, the result indicated that the system generated new noun word lexicon with 99.64% accuracy

Some incorrect nouns were shown in table 6.2.

Table 6.2 : The result of incorrect noun lexicon builder

No	Rules	Incorrect Noun
1.7	ence-NOUN	hence
1.8	er-NOUN	altogether, either, ever, forever, her, however, howsoever, moreover, neither, over, together, under, wherever, whenever
1.14	or-NOUN	for, or
1.15	ship-NOUN	amidship
1.17	ter-NOUN	after
1.21	ty-NOUN	petty

- Adjective Lexicon

Like noun lexicon training, the system used 27 heuristic rules to search for new adjectives in training sets. Words, such as noun, verb, or duplicated adjective, were discarded. Some words, such as adverb or conjunction, were checked as incorrect adjective. The remaining were added in adjective lexicon and computed statistics of each rule. The results were shown in table 6.3 below.

Table 6.3 : The result of adjective lexicon builder

No	Rules	Correct	Incorrect	% Correct
2.1	AUX-X-TO	25	1	96.15
2.2	ART-X-ADJ	69	1	98.57
2.3	IT-IS-X	49	3	94.23
2.4	able-ADJ	368	0	100

Table 6.3 : The result of adjective lexicon builder (Continue)

2.5	al-ADJ	148	0	100
2.6	ant-ADJ	89	1	98.89
2.7	ative-ADJ	89	0	100
2.8	cal-ADJ	166	0	100
2.9	ent-ADJ	202	2	99.02
2.10	ful-ADJ	112	0	100
2.11	ial-ADJ	133	0	100
2.12	ible-ADJ	90	0	100
2.13	ic-ADJ	223	0	100
2.14	ive-ADJ	77	0	100
2.15	less-ADJ	133	3	97.79
2.16	like-ADJ	25	0	100
2.17	nal-ADJ	129	0	100
2.18	ous-ADJ	392	0	100
2.19	ral-ADJ	72	0	100
2.20	tic-ADJ	193	0	100
2.21	tive-ADJ	113	0	100
2.22	ual-ADJ	43	0	100
2.23	ular-ADJ	41	0	100
2.24	ward-ADJ	28	3	90.32
2.25	wise-ADJ	5	1	83.33
2.26	??-??ed-ADJ	134	0	100
2.27	??-??ing-ADJ	31	0	100
	TOTAL	3179	15	99.53

In our experiment, the result indicated that the system generated new adjective word lexicon with 99.53% accuracy which is a little less than those of noun lexicon (99.64%).

Some incorrect adjectives are shown in table 6.4.

Table 6.4 : The result of incorrect adjective lexicon builder

No	Rules	Incorrect Adjective
2.1	AUX-X-TO	is said to
2.2	ART-X-ADJ	the program necessary
2.3	IT-IS-X	it is seen, it is done, it is therefore
2.6	ant-ADJ	meant
2.9	ent-ADJ	sent, went
2.15	less-ADJ	unless, nevertheless, nonetheless
2.24	ward-ADJ	toward, afterward, henceforward
2.25	wise-ADJ	clockwise

- Verb Lexicon

In verb lexicon training, the system used 2 heuristic rules to search for new verb pattern in training sets. Verb patterns, starting with auxiliary verb and ending with "-ed or -ing" words, were extracted and checked the correctness by user. Table 6.5 illustrated the statistics of extracted verb pattern.

Table 6.5 : The result of verb lexicon builder

No	Rules	Correct	Incorrect	% Correct
3.1	ED-ending VP	739	10	98.66
3.2	ING-ending VP	132	5	96.35
	TOTAL	871	15	98.28

In our experiment, the result indicated that the system generated new verb patterns with 98.28 % accuracy.

Some incorrect verb patterns are shown in table 6.6.

Table 6.6 : The result of incorrect verb patterns

No	Rules	Incorrect Verb Pattern
3.1	ED-ending VP	proceed, exceed, need, led, heed, indebted, indeed, succeed, sped, speed
3.2	ING-ending VP	spring, nothing, string, bring, king

After finding verb pattern, 4 heuristic rules were used to change to candidate verb form. 8 heuristic rules were used to identify true verb from candidate verb. The results of finding true verb from candidate verb were shown in table 6.7 below.

Table 6.7 : The result of finding true verb from candidate verb

No	Rules	Correct	Incorrect	% Correct
3.3	True Verb by S-ending (7 rules)	315	0	100
3.4	True Verb by word follow TO (1 rule)	268	0	100
	TOTAL	583	0	100

In our experiment, the result indicated that the system generated new verb word lexicon with 100 % accuracy.

- Grammar Lexicon

To acquire new grammar patterns from training set, we run the parsing system until it reached final stage. The unknown grammar patterns were shown and selected by the user. The user could accept or reject patterns that system generated. The accepted patterns would be added in grammar lexicon while rejected patterns would be discarded. With this process, we could collect a new grammar pattern.

6.4.2 Text pre-processor

In text pre-processor, we used lexicon in above section for tagging each word in text documents. After that, 100% word-tagging sentences were processed by location checking and phrase grouping. The results were shown in table 6.8 below.

- Location Checking

Table 6.8 : The result of location checking

No	Rules	Correct	Incorrect	% Correct
4.1	DET[V]:N	97	0	100
4.2	DET[ED,ING V]:ADJ	62	0	100
4.3	OF[V3]:ADJ	1	0	100
4.4	ADJ[V]:N	154	3	98.09
4.5	ADJ[ED-V]:ADJ	9	0	100
4.6	PP[V]:N	37	1	97.37
4.7	[V]OF:N	84	2	97.67
4.8	[V]AUX:N	15	2	88.24
4.9	[V]V3:N	1	0	100
4.10	[Capital Letter]:N	256	0	100
4.11	[ADJ]AUX:N	24	0	100
4.12	These[CONJ]:N	2	0	100
4.13	['s]:ADJ	18	0	100
4.14	DET[PP]:ADJ	1	0	100
4.15	AUX[AUX]:V1	3	0	100
4.16	TO[AUX]:V1	1	0	100
4.17	OF[CONJ]:ADJ	4	0	100
4.18	V1[V1]:N	8	1	88.89
4.19	OF[V]:N	11	0	100
4.20	AUX[Vs]:N	1	0	100
4.21	ADJ[AND]ADJ:ADJ	7	0	100
4.22	ADJ[,]ADJ:ADJ	1	0	100
4.23	ADJ[OR]ADJ:ADJ	1	0	100

Table 6.8 : The result of location checking (Continue)

4.24	[ADJ]OF:N	14	0	100
4.25	[CONJ]OF:N	2	0	100
4.26	DET[V3]:ADJ	6	0	100
4.27	OF[ED V]:ADJ	4	0	100
4.28	[Number]:N	9	0	100
	TOTAL	833	9	98.92

The result indicated that the location checking result yields 98.92 % accuracy.

Some incorrect location checking patterns are shown in table 6.9.

Table 6.9 : The result of incorrect location checking patterns

No	Rules	Incorrect
4.4	ADJ[V]:N	these cover this presents This means that
4.6	PP[V]:N	The Internet technology described above provides
4.7	[V]OF:N	to be informed of a user required of
4.8	[V]AUX:N	the switching architecture that the authors present has the work we describe should be classified
4.18	V1[V1]:N	help solve

- Phrase grouping

After performing location checking, each tagged word in sentence were grouping into phrases (noun phrase and verb phrase). Phrase grouping results were shown in the table below.

- Noun Phrase Grouping

Table 6.10 : The result of noun phrase grouping

No	Phrase Grouping Format	Correct	Incorrect	% Correct
5.1	N	499	33	93.80
5.2	N-N	134	3	97.81
5.3	ART-N	479	21	95.80
5.4	ADJ-N	366	11	97.08
5.5	ART-N-N	167	6	96.53
5.6	ART-ADJ-N	178	5	97.27
5.7	ADJ-ADJ-N	16	1	94.12
5.8	ART-ADJ-N-N	32	1	96.97
5.9	ART-ADJ-ADJ-N	13	2	86.67
5.10	ART-N-N-N	30	0	100
5.11	ADJ-ADJ-ADJ-ADJ-N	1	0	100
5.12	ADJ-N-N	48	2	96.00
5.13	ADJ-N-N-N	7	0	100
5.14	ADJ-ADJ-N-N	4	0	100
5.15	ART-ADJ-N-N-N-N	1	0	100
5.16	ART-ADJ-ADJ-ADJ-N	2	0	100
5.17	ADJ-ADJ-ADJ-N	5	0	100
5.18	N-N-N	18	1	94.74
5.19	ART-N-N-N-N	4	0	100
5.20	ART-ADJ-ADJ-N-N	1	0	100
5.21	N-N-N-N-N	1	0	100
	TOTAL	2006	86	95.71

The result indicated that the noun phrase grouping result yields 95.71 % accuracy. Most of the incorrect noun phrases came from the factor of word disambiguation. It caused a fault semantic incorrect but in control mechanism, the next step of parsing system, some noun phrases could be adjusted into correct noun phrases which automatically increases the noun phrase grouping accuracy.

- Verb Phrase Grouping

Table 6.11 : The result of verb phrase grouping

No	Rules	Correct	Incorrect	% Correct
6.1	AUX-V ED	195	0	100
6.2	AUX-V ING	16	0	100
	TOTAL	211	0	100

The result indicated that the verb phrase grouping result yields 100 % accuracy.

6.4.3 Control Mechanism

This mechanism would be used to check the correctness of text parsing. We adjusted ambiguated words and decomposed sentence into sub-sentences in order to reduce the complexity of sentence structure. Then, the preposition-noun phrase part (called PP-NP) was removed. The remaining sentence was then checked with grammar patterns in lexicon. The results were shown in table below.

- Noun-Verb Disambiguation

Table 6.12 : The result of noun-verb disambiguation

No	Rules	Correct	Incorrect	% Correct
7.1	Noun + [N-V verb] (combine with pre-position)	73	0	100
7.2	[N-V verb] + Noun (combine with post-position)	14	0	100
7.3	Noun + [N-V verb] + Noun (combine with pre- and post-position)	15	0	100
7.4	[N-V verb] (change to Noun)	26	0	100
	TOTAL	128	0	100

The result indicated that the noun-verb disambiguation result yields 100 % accuracy.

- Sentence Decomposition

Table 6.13 : The result of sentence decomposition

No	Rules	Correct	Incorrect	% Correct
8.1	Conj-Noun	354	0	100
8.2	Conj-Pronoun	38	0	100
8.3	Conj-VP	24	0	100
8.4	Conj-V1	26	0	100
8.5	Conj-Vs	22	0	100
8.6	Conj-Aux	32	0	100

Table 6.13 : The result of sentence decomposition (Continue)

8.7	Conj-Ving	16	0	100
8.8	Conj-Ved	15	0	100
8.9	Noun-V3	9	0	100
8.10	Noun-Ved	40	0	100
8.11	Noun-Ving	13	0	100
8.12	Noun-Aux	72	0	100
8.13	Noun-VP	129	0	100
8.14	Noun-Noun	21	0	100
8.15	Noun-Pro	7	0	100
	TOTAL	818	0	100

The result indicated that the sentence decomposition result yields 100 % accuracy.

- PP-NP Removal

Table 6.14 : The result of PP-NP removal

No	Rules	Correct	Incorrect	% Correct
9.1	[PP-NP]	637	0	100
9.2	[PP-NP]-PP	184	0	100
9.3	[PP-NP]-To	31	0	100
	TOTAL	852	0	100

The result indicated that the preposition-noun phrase removal result yields 100 % accuracy.

- Grammar Checking

There are 166 generated grammar patterns in grammar lexicon. The result of grammar matching was illustrated in table 6.15.

Table 6.15 : The result of grammar patterns

No	Grammar Pattern	Match
10.1	NP	356
10.2	NP-VP	158
10.3	NP-AUX-NP	85
10.4	NP-Vs-NP	53
10.5	NP-Ved	47
10.6	NP-AUX-ADJ	39
10.7	NP-AUX-V1-NP	37

Table 6.15 : The result of grammar patterns (Continue)

10.8	NP-V1-NP	35
10.9	NP-AUX	21
10.10	NP-Ving-NP	20
10.11	NP-AUX-V1	16
10.12	PRO-VP	14
10.13	NP-AUX-ADJ-TO-NP	13
10.14	NP-V1	13
10.15	NP-VP-TO-NP	13
10.16	NP-VP-TO-V1-NP	13
10.17	NP-VP-NP	12
10.18	NP-PP-Ving-NP	11
10.19	NP-Vs	11
10.20	NP-TO-NP	9
10.21	NP-TO-V1-NP	9
10.22	PRO-V1-NP	9
10.23	NP-Ved-NP	8
10.24	NP-V3	7
10.25	NP-AUX-TO-V1-NP	6
10.26	NP-Ved-TO-V1-NP	6
10.27	PRO-AUX-ADJ	6
10.28	NP-ADJ	5
10.29	PRO-AUX-V1-NP	5
10.30	TO-V1-NP	5
10.31	NP-AUX-NP-TO-V1- NP	4
10.32	NP-PP-NP-Vs	4
10.33	NP-PP-NP-Vs-NP	4
10.34	NP-Ving	4
10.35	NP-VP-ADJ	4
10.36	NP-Vs- NP-PP-Ving- NP	4
10.37	PP	4
10.38	PRO-AUX	4
10.39	PRO-AUX-ADJ-TO-V1	4
10.40	PRO-AUX-NP	4
10.41	PRO-V1	4
10.42	V1	4
10.43	Ving-NP	4
10.44	NP-AUX-ADJ-PP-Ving-NP	3
10.45	NP-AUX-V1-TO-V1- NP	3
10.46	NP-PP-NP-V1	3
10.47	NP-TO-V1	3
10.48	NP-V3-TO-NP	3
10.49	NP-Ved-TO-NP	3

Table 6.15 : The result of grammar patterns (Continue)

10.50	NP-Ving-NP-TO-V1-NP	3
10.51	NP-VP-ADJ-TO-NP	3
10.52	NP-VP-PP	3
10.53	NP-Vs- NP- TO-V1- NP	3
10.54	NP-Vs-NP-TO-NP	3
10.55	PRO-AUX-ADJ-TO-V1-NP	3
10.56	NP-AUX-ADJ-TO-V1	2
10.57	NP-AUX-ADJ-TO-V1- NP	2
10.58	NP-AUX-NP-PP-PRO	2
10.59	NP-AUX-NP-TO-NP	2
10.60	NP-AUX-PP	2
10.61	NP-AUX-TO-NP	2
10.62	NP-AUX-V1-TO-NP	2
10.63	NP-AUX-V1-TO-V1	2
10.64	NP-PP-NP-V1-NP	2
10.65	NP-V1-TO-V1-NP	2
10.66	NP-Ved-PP	2
10.67	NP-Ving-PRO	2
10.68	NP-Ving-TO-V1-NP	2
10.69	NP-VP-PP-Ving-NP	2
10.70	NP-Vs-NP-TO-V1	2
10.71	NP-Vs-NP-TO-V1-NP-TO-NP	2
10.72	PRO-Ved	2
10.73	PRO-Vs	2
10.74	PRO-Vs-NP	2
10.75	PRO-Vs-NP-TO-NP	2
10.76	TO-NP	2
10.77	NP-AUX- NP-PP-Ving- NP	1
10.78	NP-AUX-ADJ-PP	1
10.79	NP-AUX-ADJ-PP-ADJ- NP	1
10.80	NP-AUX-ADJ-PP-Ving-TO-NP	1
10.81	NP-AUX-ADJ-TO	1
10.82	NP-AUX-ADJ-TO-V1-TO-NP	1
10.83	NP-AUX-ADV-NP	1
10.84	NP-AUX-NP-PP	1
10.85	NP-AUX-NP-PP-Ving	1
10.86	NP-AUX-NP-PP-Ving-NP-ADJ-TO-NP	1
10.87	NP-AUX-NP-TO-V1	1
10.88	NP-AUX-PP-TO-NP	1
10.89	NP-AUX-TO-V1- NP-PP-ADJ	1
10.90	NP-AUX-TO-V1-NP-PP-ADJ-NP	1
10.91	NP-AUX-TO-V1-NP-PP-Ving-NP	1

Table 6.15 : The result of grammar patterns (Continue)

10.92	NP-AUX-TO-V1-TO- NP	1
10.93	NP-AUX-TO-V1-Ved-NP-TO-NP	1
10.94	NP-AUX-V1-ADJ	1
10.95	NP-AUX-V1-ADJ-NP	1
10.96	NP-AUX-V1-NP-PP-Ving-NP	1
10.97	NP-AUX-V1-NP-TO-V1-NP	1
10.98	NP-AUX-V1-NP-TO-V1-NP-TO-NP	1
10.99	NP-AUX-V1-PP	1
10.100	NP-AUX-V1-PP-Ving- NP	1
10.101	NP-AUX-V1-Ving	1
10.102	NP-PP	1
10.103	NP-PP-PRO	1
10.104	NP-PP-Ving	1
10.105	NP-TO-V1-NP-PP	1
10.106	NP-TO-V1-PRO	1
10.107	NP-V1-ADJ	1
10.108	NP-V1-ADJ-TO-NP	1
10.109	NP-V1-PP-TO-NP	1
10.110	NP-V1-TO-NP	1
10.111	NP-V3-TO-Ving-NP	1
10.112	NP-Ved-ADJ	1
10.113	NP-Ved-NP-TO-V1-NP	1
10.114	NP-Ved-PP-NP-Vs- NP	1
10.115	NP-Ved-PP-Ving	1
10.116	NP-Ved-TO-V1-NP-TO-NP	1
10.117	NP-Ving-ADJ-NP	1
10.118	NP-Ving-TO-NP	1
10.119	NP-VP-ADJ-PP	1
10.120	NP-VP-ADJ-PP-Ving-NP	1
10.121	NP-VP-AUX-NP	1
10.122	NP-VP-NP-PP-Ving	1
10.123	NP-VP-NP-TO-V1- NP	1
10.124	NP-VP-PP-Ved-NP	1
10.125	NP-VP-PP-Ving	1
10.126	NP-VP-PP-Ving-NP-TO-NP	1
10.127	NP-VP-TO	1
10.128	NP-VP-TO-PRO	1
10.129	NP-VP-TO-V1-ADJ	1
10.130	NP-VP-TO-V1-PP-PRO	1
10.131	NP-Vs- NP-PP- NP-ADJ	1
10.132	NP-Vs-NP-TO-V1-PP	1
10.133	NP-Vs-PP-NP-ADJ-TO-NP	1

Table 6.15 : The result of grammar patterns (Continue)

10.134	NP-Vs-PP-Ving-NP	1
10.135	NP-Vs-TO-V1	1
10.136	NP-Vs-TO-V1-NP	1
10.137	NP-Vs-TO-V1-PP-PRO	1
10.138	NP-Vs-V1-NP	1
10.139	PP-ADJ	1
10.140	PP-NP-Vs-PP-Ving-NP	1
10.141	PRO-AUX-ADJ-PP	1
10.142	PRO-AUX-ADJ-PP-ADJ	1
10.143	PRO-AUX-ADJ-TO-V1-NP-TO-NP	1
10.144	PRO-AUX-NP-TO-V1	1
10.145	PRO-AUX-TO-NP	1
10.146	PRO-AUX-V1	1
10.147	PRO-AUX-V1- NP-TO-V1- NP	1
10.148	PRO-AUX-V1-TO-NP	1
10.149	PRO-NP	1
10.150	PRO-V1-ADJ	1
10.151	PRO-V1-NP-PP-Ving-NP-TO-NP	1
10.152	PRO-V1-TO-V1-NP	1
10.153	PRO-V1-TO-V1-NP-ADJ-TO-V1-NP	1
10.154	PRO-V1-TO-V1-NP-PP-Ving-NP	1
10.155	PRO-Ved-NP	1
10.156	PRO-VP-ADJ	1
10.157	PRO-VP-NP-TO-V1-PP-ADJ	1
10.158	PRO-VP-PRO-ADJ	1
10.159	PRO-VP-TO-NP	1
10.160	NP-VP-TO-V1	1
10.161	PRO-VP-TO-V1	1
10.162	PRO-Vs- NP-TO-V1- NP	1
10.163	PRO-Vs-PP-Ving-NP	1
10.164	PRO-Vs-TO-PRO	1
10.165	Ved	1
10.166	Ving-NP-TO-V1-NP	1

6.4.4 Unknown Lexicon

In case of unknown words that not included in lexicon, we could classify them by using machine learning technique with 8 heuristic rules (4 rules to select unknown words, 3 rules to classify noun, 1 rule to classify adjective). After performing phrase grouping step, the unknown words in each pattern that matched with defined heuristic rules were extracted and added into lexicon.

- Heuristic Rules to classify unknown words as Noun

Table 6.16 : The result of classifying unknown words as noun

No	Rules	Correct	Incorrect	% Correct
11.1	[ADJ-X] [NP-X]	47	0	100
11.2	[NP-X] [X-NP]	36	0	100
11.3	[ART-X] [NP-X]	37	0	100
	TOTAL	120	0	100

The experimental results indicated that the system generated new noun word lexicon from unknown words with 100 % accuracy.

- Heuristic Rules to classify unknown words as Adjective

Table 6.17 : The result of classifying unknown words as adjective

No	Rules	Correct	Incorrect	% Correct
12.1	[X-ADJ] [X-NP]	16	0	100

The experimental results indicated that the system generated new adjective word lexicon from unknown words with 100 % accuracy.

6.4.4 The Result of Extracting Noun Phrases from Sentence

The extracted noun phrases were presented to a user for manual review. The overall efficiency of the TEPT parsing system is shown in table 6.18 below.

Table 6.18 : The efficiency of TEPT parsing system

	Sentence	Mean Word	Extracted NP	Correct	Incorrect	% Correct
Simple	218	10.9	675	673	2	99.7
Compound	91	17.8	420	418	2	99.5
Complex	208	19.0	1028	1023	5	99.5
TOTAL	517	15.4	2123	2114	9	99.6

The result indicated that the parsing result yields 99.6 % accuracy.

CHAPTER VII

CONCLUSION

7.1 Conclusion

In the previous chapter, we have bestowed TEPT concept that uses heuristic rules and machine learning in lexicon acquisition and parsing texts. We also exploited three distinct features of TEPT which are its dynamic, learnable and portable properties. New specific domain lexicons will be easily acquired by training TEPT with a set of that particular domain texts. Therefore, these features are appropriate to the current electronic document flooding problems on Internet.

From the experiments, we can conclude about TEPT's performance in recall-precision aspects. The term "recall" means how much correct information was selected. The second term "precision" means how much of the information selected was correct. The recall-precision results are presented into four tables below.

Table 7.1 shows Recall-Precision results of lexicon builder

Lexicon Builder	Recall	Precision	
	Word	Word	(%)
Noun Lexicon	5652	5632	99.64
Adjective Lexicon	3195	3180	99.53
Verb Lexicon			
- select verb pattern	871	856	98.28
- find true verb by s-ending	315	315	100
- find true verb by word follow TO	268	268	100

Table 7.2 shows Recall-Precision results of text pre-processor

Text Pre-processor		Recall	Precision	
		W/P	W/P	(%)
Location Checking	(word)	833	824	98.92
Noun Phrase Grouping	(phrase)	2006	1920	95.71
Verb Phrase Grouping	(phrase)	211	211	100

Table 7.3 shows Recall-Precision results of control mechanism

Control Mechanism		Recall	Precision	
		W/P	W/P	(%)
Noun-Verb Disambiguation	(word)	128	128	100
Sentence Decomposition	(pattern)	818	818	100
PP-NP Removal	(pattern)	852	852	100

Table 7.4 shows Recall-Precision results of parsing result

Parsing Result		Recall	Precision	
		W/P	W/P	(%)
Noun Phrases Extracted	(phrase)	2123	2114	99.6
Domain Word	(word)	917	538	58.67

In parsing result, the number of processed sentences is about 1500 sentences. TEPT selects only 100% tagged-word sentence for parsing because it makes the best performance in accuracy parsing result. The selected sentences are 517 sentences (34.47 % recall). The percentage of tagged-word sentence controls the number of selected sentence for parsing. Lower percentage means higher unknown words. To gain higher percentage of recall, we should test TEPT with a variety of threshold settings, for example, from 90% to 100%. But the users may be awared of the correctness of parsing result.

The experimental results shown in table 7.1, 7.2, 7.3 and 7.4 indicate that the heuristic rules approach with machine learning, used in TEPT, are successful in each routine operation (lexicon builder, text pre-processor, and control mechanism), especially in automated learnable lexicon, grammar. We demonstrated that the approach of TEPT is effectively used to extract noun phrases which are main idea in the texts with high accuracy (99%). Among the extracted noun phrases, 58% noun phrases were found as relevant keywords when compare with Computer Science domain. Moreover, TEPT can be portable to other domain based on training corpus. As a result, we can conclude that TEPT is an appropriate tool for domain specific resources acquisition and text parsing tool for any specialists in different domains.

7.2 Future Work

The followings are interesting directions for future work.

1) The complex noun phrases or proper name recognition may be added to improve the learning performance and to reduce the incorrect interpretation in parsing.

2) The lexicon must be sense-based rather than word-based. The semantic part may be constructed into domain lexicon.

3) To cover more new words in lexicon acquisition and to control better parsing performance, the new heuristic rules may be added into TEPT.

4) TEPT will be tested on other domain.

REFERENCES

1. Chute CG, Yang Y. An example-based mapping method for text categorization and retrieval. ACM Transactions on Information Systems 1994;12(3):252-77.

2. Cowie J, Lehnert W. Information extraction. Communications of The ACM 1996;39(1):80-91.

3. David C. The TreeBanker: a tool for supervised training of parsed corpora. ACL Workshop: Computational Environments for Grammar Development and Linguistic Engineering (Madrid): Cambridge (UK); SRI International 1997, Available from: URL: http://www.cam.sri.com/tr/crc068/paper.ps.Z [Accessed 1996 Sep 12].

4. Fred K. Constraint grammar as a framework for parsing unrestricted text. Proceedings of the 13th International Conference of Computational Linguistics, Vol. 3. Helsinki, 1990:168-73.

5. Guthrie L. The role of lexicons in natural language processing. Communications of The ACM 1996;39(1):63-72.

6. Humphrey P. Natural language processing at EDS. EDS Research Internal; 1992.

7. Jacobs PS, Rau LF. SCISOR: extracting information from on-line news. Communication of the ACM 1990;33(11):88-97.

8. Jacobs PS, Rau LF. Innovations in text interpretation. Artificial Intelligence 1993;63:143-91.

9. Kenneth CW, Lisa FR. Commercial applications of natural language processing. Communication of The ACM 1995;38(11):71-9.

10. Koymen K. TOS: a text organizing system. Information Processing & Management 1975;11:23-38.

11. Krovetz R, Croft WB. Lexical ambiguity and information retrieval. ACM Transactions on Information Systems1992April;10(2):115-41.

12. Langley P, Simon Herbert A. Applications of machine learning and rule induction. Communication of the ACM 1995;38(11):55-64.

13. Lehnert W. Symbolic/subsymbolic sentence analysis: exploiting the best of two worlds. In: Barnden J, Pollack J, editors. Advances in Connectionist and Neural Computation Theory; 1991(1). Norwood (NJ): Ablex Publishers; 1991. p. 135-64.

14. Liddy ED, Paik W, Yu ES. Text categorization for multiple users based on semantic features from a machine-readable dictionary. ACM Transactions on Information Systems 1994;12(3):278-95.

15. Rich E, Knight K. Artificial intelligence. 2nd ed. New York: McGraw-Hill; 1991.

16. Riloff E. Automatically generating extraction patterns from untagged text. Proceedings of the Thirteenth National Conference on Artificial Intelligence; AAAI Press/The MIT Press; 1996.

17. Riloff E, Lehnert W. High-precision text classification. ACM Transactions on Information Systems 1994July;12(3):296-332.

18. Salton G. Automatic text analysis. Science 1970Apr;168(3929):335-43.

19. Salton G. Developments in automatic text retrieval. Science 1991Aug;253:974-9.

20. Swan M. Practical english usage. Oxford: Oxford University Press; 1980.

21. Turban E. Decision support and expert systems : management support systems. 3rd ed. New York: Macmillan; 1993.

22. Tzeras K, Hartmann S. Automatic indexing based on Bayesian inference Networks. ACM-SIGIR 1993:22-34.

BIOGRAPHY

NAME Mr. Kullawat Wuttisunkornsakul

DATE OF BIRTH 3 November 1961

PLACE OF BIRTH Lampang, Thailand

INSTITUTIONS ATTENDED

Chiengmai University, 1979-1984

Bachelor of Pharmacy

Mahidol University, 1991-1992

Bachelor of Science (Computer Science)

POSITIONS HELD & OFFICE

1989 Non-GPO Product Warehouse Section,
Non-GPO Product Logistics Division,
Logistics Department,
Government Pharmaceutical Organization,
Bangkok, Thailand.

1999 Central Distribution Section,
GPO Product Logistics Division,
Logistics Department,
Government Pharmaceutical Organization,
Bangkok, Thailand