METASEARCH ENGINE USING FUZZY LOGIC AND CONTENT-BASED CLUSTERING APPROACHES

WERASAK YUOKOOLBODEE

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR
THE DEGREE OF MASTER OF SCIENCE
(COMPUTER SCIENCE)
FACULTY OF GRADUATE STUDIES
MAHIDOL UNIVERSITY
2004

ISBN 974-04-5096-2

Thesis
Entitled

METASEARCH ENGINE USING FUZZY LOGIC AND CONTENT-BASED CLUSTERING APPROACHES

……………………………….
Mr. Werasak Yuokoolbodee
Candidate

……………………………….
Asst.Prof. Chomtip Pornpanomchai, Ph.D.
Major-Advisor

……………………………….
Assoc.Prof. Damras Wongsawang, Ph.D.
Co-Advisor

……………………………………………………
Assoc.Prof. Rassmidara Hoonsawat, Ph.D.
Dean
Faculty of Graduate Studies

……………………………………………………
Assoc.Prof. Supachai Tangwongsan, Ph.D.
Chair
Master of Science (Computer Science)
Major in Computer Science
Faculty of Science

Thesis
Entitled

METASEARCH ENGINE USING FUZZY LOGIC AND CONTENT-BASED CLUSTERING APPROACHES

was submitted to the Faculty of Graduate Studies, Mahidol University
For the degree of Master of Science (Computer Science)
on
30 July, 2004

ACKNOWLEDGEMENTS

I am enormously grateful to my thesis advisor Asst.Prof. Chomtip Pornpanomchai for his valuable advice, generosity and encouragement throughout which enable myself to achieve this thesis.

I would like to thank Assoc.Prof. Damras Wongsawang, Asst.Prof. Sukanya Phongsuphap and Dr. Panjai Tantatsanawong, my committee members, for their valuable suggestion and their attention.

I would like to thank Assoc.Prof. Supachai Tangwongsan; chairman of the M.Sc.course; Major in Computer Science for valuable advice and guidance in this research.

I would like to give special thank to the following people who provided various forms of assistance which enabled me to achieve this thesis: Dr. Michael A. Allen and Miss Suwanna Dunyanukit for their assistance in examining this research; the thirteen volunteer, for their valuable advice, encouragement and wonderful friendships.

Finally, I am grateful to my parents and my uncle for their financial support, encouragement, and love. The usefulness of this thesis, I dedicate to my father, my mother and my sister who have inspired me to achieve my goal.

Werasak Yuokoolbodee

METASEARCH ENGINE USING FUZZY LOGIC AND CONTENT-BASED CLUSTERING APPROACHES

WERASAK YUOKOOLBODEE 4337420 SCCS/M

M.Sc.(COMPUTER SCIENCE)

THESIS ADVISORS : CHOMTIP PORNPANOMCHAI Ph.D., DAMRAS WONGSAWANG Ph.D.

ABSTRACT

Due to the enormous data on the internet it is very difficult for the web users to find the information they want. Search engines have been created to solve the problem by matching the query with the entry data in the database and retrieving the information for web users. However, the amount of data on the internet is still increasing rapidly. Gathering all available data on the internet and keeping it up to date is very hard for a single search engine. Therefore, web developers have developed a system that can send a query to several search engines at one time and integrate the results into one list. The system is called a metasearch engine. Its features enhance time saving, increase searching coverage and help users to find relevant information more quickly than using several single search engines.

Normally, in searching data, not all terms are equally important in a query. Thus, it is important to allow users to indicate the relative importance of various terms by weighting them. It is noteworthy that weighting may have an impact on the entire set of retrieved document. This thesis proposes the use of a fuzzy logic ranking. The fuzzy logic ranking will retrieve the information followed by terms that web users can weight by themselves. Furthermore, the ranking mechanism works well when web users can formulate a well-defined query for searches. However, a lot of users formulate very short queries and are unfamiliar with the topics they are looking for, therefore, most results satisfy their needs. In solving such a problem, a content-based clustering (CBC) technique is considered very helpful as it will divide the results into several groups, which may be matched to the areas of the users interest.

CBC aims to yield the clusters that can fulfill a precision of results. In addition, a prototype of a metasearch engine using fuzzy logic and content-based clustering was developed as proof of the proposed concept. Although the system takes several minutes to retrieve all of the results pages, the experiments show that this system outperforms other systems.

KEY WORDS : METASEARCH ENGINE / FUZZY LOGIC /CLUSTERING/
INFORMATION RETRIEVAL/

197 pp. ISBN 974-04-5096-2

เครื่องมือค้นหาแบบเมตาเสริช โดยใช้ตรรกศาสตร์คลุมเครือและการจัดกลุ่มข้อมูลด้วยเนื้อหาของข้อมูล (METASEARCH ENGINE USING FUZZY LOGIC AND CONTENT-BASED CLUSTERING APPROACHES)

วีรศักดิ์ เหย่ากุลบดี 4337420 SCCS/M

วท.ม. (วิทยาการคอมพิวเตอร์)

คณะกรรมการควบคุมวิทยานิพนธ์ : ชมทิพ พรพนมชัย, Ph.D., ดำรัส วงศ์สว่าง, Ph.D.

บทคัดย่อ

เนื่องจากจำนวนข้อมูลที่มีมากบนอินเตอร์เนตมันจึงเป็นเรื่องยากสำหรับผู้ใช้ในการค้นหาข้อมูลที่ต้องการ Search Engine จึงถูกสร้างขึ้นมาเพื่อแก้ปัญหาโดยทำเปรียบเทียบระหว่างคำที่ต้องการค้นหากับข้อมูลทั้งหมดที่เก็บในฐานข้อมูลและส่งผลที่ได้กลับไปยังผู้ใช้ อย่างไรก็ตามจากจำนวนข้อมูลบนอินเตอร์เนตที่เพิ่มขึ้นอย่างรวดเร็วส่งผลให้การเก็บรวบรวมข้อมูลบนอินเตอร์เนตและปรับปรุงให้ทันสมัยอยู่เสมอนั้นเป็นเรื่องยากเมื่อใช้ Search Engine เพียงตัวเดียว ดังนั้นนักพัฒนาด้านเวปจึงพัฒนาระบบที่สามารถส่งคำที่ต้องการค้นหาไปยังหลายๆ Search Engine เพียงครั้งเดียวและรวบรวมผลที่ได้ทั้งหมดเป็นรูปแบบเดียวกัน ระบบนี้เรียกว่า Metasearch Engine ซึ่งจุดเด่นของระบบนี้คือลดเวลาในการรับส่งข้อมูล, ครอบคลุมการค้นหาข้อมูล และช่วยผู้ใช้ในการค้นหาข้อมูลที่เกี่ยวข้องได้ดีกว่าการใช้ Search Engine หลายๆตัว

โดยทั่วไปในการค้นหาข้อมูล คำทุกคำที่ใช้ในการค้นหานั้นจะมีความสำคัญไม่เท่าเทียมกัน ดังนั้นการอนุญาตให้ผู้ใช้กำหนดน้ำหนักของคำที่ต้องการหาจึงเป็นเรื่องสำคัญ เพราะน้ำหนักที่กำหนดจะมีผลกระทบกับผลลัพธ์ทั้งหมดที่จะได้ ในวิทยานิพนธ์นี้นำเสนอการจัดลำดับข้อมูลโดยใช้ตรรกศาสตร์คลุมเครือ การจัดลำดับข้อมูลจะทำการเรียกค้นข้อมูลตามน้ำหนักของแต่ละคำที่ผู้ใช้กำหนดด้วยตนเอง นอกจากนี้กระบวนการการจัดลำดับของข้อมูลจะทำได้ดีก็ต่อเมื่อผู้ใช้กำหนดคำที่ต้องการค้นหาได้ดี อย่างไรก็ตามผู้ใช้ส่วนใหญ่จะใช้คำสั้นๆในการค้นหาและก็ไม่คุ้นเคยกับหัวข้อที่ต้องการค้นหาทำให้ผลลัพธ์ส่วนใหญ่ที่ได้ไม่ตรงกับความต้องการ ในการแก้ไขปัญหาวิธีการจัดกลุ่มข้อมูลด้วยเนื้อหาของข้อมูลจะช่วยได้มาก โดยจะทำการจัดแบ่งข้อมูลออกเป็นหลายๆกลุ่ม ซึ่งทำให้ตรงกับความสนใจของผู้ใช้

การจัดกลุ่มข้อมูลด้วยเนื้อหาของข้อมูลมีจุดมุ่งหมายเพื่อให้ได้ผลลัพธ์ที่ถูกต้อง นอกจากนี้ยังได้สร้างต้นแบบเครื่องมือค้นหาแบบเมตาเสริช โดยใช้ตรรกศาสตร์คลุมเครือและการจัดกลุ่มข้อมูลด้วยเนื้อหาของข้อมูลเพื่อพิสูจน์แนวความคิดของวิทยานิพนธ์นี้ด้วย แม้ว่าระบบนี้จะใช้เวลาหลายนาทีในการเรียกค้นข้อมูลทั้งหมด แต่จากผลการทดลองแสดงให้เห็นว่าระบบนี้ทำงานได้ดี

197 หน้า ISBN 974-04-5096-2

CONTENTS

CONTENTS (CONT.)

CONTENTS (CONT.)

CONTENTS (CONT.)

LIST OF TABLES

LIST OF TABLES (CONT.)

LIST OF FIGURES

LIST OF FIGURES (CONT.)

CHAPTER 1
INTRODUCTION

1.1 Motivation

The internet is made up of millions of computers linked together around the world in such a way that it can provide a range of services to communicate and share information. Internet users can send multimedia data (e.g. image, audio, video, etc.) to colleagues and friends by using electronic mail, internet telephone and play interactive multimedia games with their friends. Although the boom in the use of the web and its very fast growth are now well known (as of June 2002, the internet contained an estimated 200 millions hosts and 800 millions users [1]). This enormous amount of data becomes very difficult for the new web users to find the document they want related to particular information they are searching for.

Due to the problems mentioned above, web developers have created a search engine to solve the problems. Each search engine has its own database to match the query with the entries in the database and retrieve the information for web users. Consequently, it has become the most popular tool for searching information the on internet since 85% of web users surveyed claim to use search engines or some kind of search tool to find specific information of interest [2].

However, the amount of data in the internet is increasing rapidly. Gathering all data in the Internet and keeping them up-to-date are very difficult, even the large search engines like google, alta vista, hotbot, northern light and excite. According to the studies, the number of indexes of these engines covered 28-55%[3] or 14-34%[4] of all web pages. That means each of the search engines has a limitation of the number of indexes.

The limitation of index information which does not cover all web pages is a time consuming process when web users search for the information they need. As they will to each search engine which has a searching service, then send the query to each search engine and make a decision for each result to be relevant to their interest. Besides, there are many public search engines that web users are not satisfied with, such as different formats for inputting queries, speeds of retrieval, presentation formats of the retrieval results and quality of retrieved information.

In solving the above problems, web developers have created a system that can send your search query to several search engines at one time and integrates the results into one list. This means users do not have to spend much time going to each individual site. The system is called a metasearch engine.

A metasearch engine has to combine the results of many sources in unified result, enhancing time saving and searching coverage. This feature helps users to find relevant information how one search engine rather than using several single search engines. An outcome of relevant information returned from the metasearch engine will depend on the ranking algorithm and other techniques.

1.2 Problem statement

As mentioned earlier, the web search engine is the most commonly used tool for information retrieval on the web; however, its current status is far from satisfactory for several possible reasons [5]:

1. Very limited coverage of the web.
2. Out of date pages and broken hyperlinks.
3. Many useful or relevant pages are not returned.
4. Many returned pages are useless or irrelevant.

5. Users may be just interested in “more qualified” information or a small part of information returned while thousands of pages are returned from search engine.
6. Different users have different requirements and expectations for search results.
7. Sometimes search requests can not be expressed clearly just in several keywords.
8. Different words have similar meanings and same words have different meanings that make things more complicated.

Due to the problems mentioned above, a metasearch engine has been developed to resolve the disadvantages of search engines by enhancing time saving, search coverage and uniform result presentation. Usually, a metasearch engine is just an interface. It does not maintain its own index on documents. When a metasearch engine receives a user query, it first passes the query to the appropriate local search engines, and then collects the results from its local search engine and present the result in one uniform format. However, a sophisticated metasearch engine may maintain information in a database to provide a better service [6]. The thesis has provided an index for all documents in its own database that is created and stored in this metasearch engine to speed up query processing.

After a searching process, the metasearch engine will pull the most relevant information to the top list of search results, ordered by their estimated relevance to the query. The relevant information is estimated based on the similarity between the text of a document and the query. Therefore, the metasearch engine should use appropriate ranking algorithm to avoid low precision searches.

The information retrieval research community has continued to develop many models for the ranking technique over the last 30 years. The ranking technique depends on:

- Relevance ranking : the location and frequency of keywords.
- Frequency ranking : how many times the keywords appear in the web page and where they appear (i.e. in the title, META description, first paragraph, rest of the main body).

Before ranking a query to a set of documents, the web developer should consider which type of ranking models are suitable for the retrieval purpose. In this research approach, *fuzzy ranking* is proposed to retrieve the information followed with terms that web users can weight within themselves. Not all terms are equally important in a query. Thus, it is significant to allow users to indicate the relative importance of various terms by weighting them. However, too much weighting may affect the entire set of retrieved document [7].

The ranking mechanism works well when web users can formulate a well-defined query for searches. However, a lot of users formulate very short queries (70% are single word queries [8]). And there are large documents retrieved for analysing. Such problems are exacerbated when the users are unfamiliar with the topics that they are looking for. Because they are novices at performing searches.

Due to the problem above, a clustering technique would help a lot. The clustering divides the results into several groups, which may match the areas of the users interest. If web search results can be presented in groups, users could have an overview of the whole topic or just select interesting groups to browse. This thesis, therefor proposes *content-based clustering(CBC)* which aims to yield the clusters that fulfill a precision of results. CBC uses the contents of each web page document to be clustered into groups. The system has to extract only the important words in the web page document to represent the whole document to reduce the noise of the data. Numerous document clustering algorithms prefer to use off-line searching but easy-to-browse result presentation than the on-line searching but hard-to-find relevant results.

The thesis also employs a keyphrase extraction to assist fuzzy ranking and content-based clustering. The keyphrase extraction provided important means of document summarization, clustering, and topic search. Such a method gives a high-level description of a document's content, of which important keywords, topical phrases are selected from within the body of the document. *KEA* – a practical automatic keyphrase extraction, is used because of its highest efficiency [9]. In English term query, the system uses *KEA* to extract important keywords in homepage contents. However, *KEA* does not support THAI language. So the system uses the *boost-up ordering algorithm* to ranking THAI term query.

Although the system takes several minutes to retrieve all of the result pages, it should use an appropriate technique to yield the most useful information to present to the web users.

1.3 Scope of thesis

The intended purpose of this thesis is to develop a prototype of a metasearch engine using the fuzzy logic to retrieve information to satisfy users need on which they can define weighting of keywords by themselves, with its terms joined together by logical connectives. Typically, the permitted connectives are *AND* and *OR*. Also, to increase effectiveness of metasearch engine, the system has its own database to develop a post-retrieval clustering called c*ontent-based clustering (CBC*) technique to help users divide the returned results of search engine easier and digest the relationship inside a group and among other groups. The system uses keyphrase extraction to assist in English term searching and *boost-up ordering algorithm* in Thai term query, as well as evaluate the technique and the prototype.

1.4 Objectives

The objectives of this thesis are to:

1. Study information retrieval, search engines, metasearch engine, fuzzy model, ranking algorithm, clustering algorithm.

2. Analyze the existing problems of metasearch engine.

3. Design the fuzzy reranking and clustering algorithm called content-based clustering (CBC).

4. Develop the metasearch engine using fuzzy logic and content-based clustering (MEUFLACC) prototype.

5. Evaluate approach and prototype.

1.5 Organization of the thesis

The rest of this thesis is organized as follows:

Chapter II reviews the concepts and current research related to this works such as the information retrieval concepts, search engine and metasearch engine approaches. This chapter also describes the basic concept of keyphrase extraction (KEA).

Chapter III presents the proposed approaches and introduces the fuzzy model, content-based clustering and keyphrase extraction tool.

Chapter IV describes the experimental environment and all details of the approaches that are presented step by step in structure chart, data flow diagram (DFD), flow chart and also illustrates the user interface of the prototype.

Chapter V shows experimental results and compare relevant ranking between the approaches and other techniques such as other search engine. This part also shows the effect factors on the approaches such as speed of internet and traffic of each web site.

Chapter VI presents the discussion of the proposed model in many points of view such as advantages, disadvantages and limitation compared with other techniques. This section also discusses the future work and research direction in a continuation of the approaches.

CHAPTER 2
LITERATURE REVIEW

This chapter aims to review previous work with a focus on the fields of information retrieval; describe the concept of information retrieval, fuzzy logic, search engine and metasearch engine as well as to explain the advantage of metasearch engine. It also explains a clustering algorithm, keyphrase extraction and comparison search engine.

It first reviews the basic concept of information retrieval. Next, it describes the process of fuzzy logic, search engine metasearch engine, clustering algorithm and keyphrase extraction. Finally, it explains the comparison search engine.

2.1 Information retrieval

Information retrieval (IR) has become accepted as a description of work published by Cleverdon, Salton, Sparck Jones, Lancaster and others. However, a perfect definition along these lines below is given by Lancaster [10]: “Information retrieval is the term conventionally, though somewhat inaccurately, applied to the type of activity discussed in this volume. An information retrieval system does not inform (i.e. change the knowledge of) the user on the subject of his inquiry. It merely informs on the existence (or non-existence) and where about of documents relating to his request”.

Information retrieval deals with the representation, storage, organization and access to information items. The representation and organization of the information items should provide the users with easy access to the information in which they are interested. However, IR was seen as a narrow area of interest mainly to librarians and information experts and there has been a change in an expansion of the World Wide Web (WWW). Nowadays, research in IR includes modeling, document classification and categorization, user interfaces, data visualization, filtering, etc.

2.1.1 Differences from older technologies

IR is regarded as a narrow area of interest mainly to librarians and information experts. Until earlier 1990, the growth of internet and web has been spreading rapidly since. The web becomes the universe of human knowledge and culture which can be compared with more traditional information dissemination technologies such as books, journals, newspapers, radio, and television. The web features the following properties [11]:

1. The web offers easily access of published information for everyone that can uses computer. The web users can easily post the information on a web server, requiring no external approval, and mostly free of charge. By lowering the barrier to entry, this has set up diverse information providers. As a result, there is an enormous amount of information available to Internet users and this amount continues to grow fast. It also makes available information more diverse in topic, presentation, style, quality, purpose, etc. In particular, the diversity in quality means that there is almost no guarantee for the quality of any available piece of information. This makes information selection much harder for the consumers.

2. The web greatly simplifies the process of information retrieval. The web users just has to connect to the right web server (an information resource once its location is known) and request the appropriate page that they are looking for. The process is easy, fast and free of charge in most cases, and also not censored in any subject. As a result, information users have a greater variety of information objects to consume or examine concurrently with many more choices for them to select.

3. Information resources on the web are so changeable that the information provider can be modified or removed easily. Hence, many web resources are more dynamic and kept up-to-date. While some other resources quickly become outdated and rather limited. Currently, it is still difficult to derive structured data or accurate information from HTML pages on the web.

2.1.2 Retrieval evaluation of IR

In measuring the usefulness of a IR system, it can be evaluated by several methods such as speed of a IR system, the required space of a IR system, style of presentation of information to user, etc. However, the mostly used method in IR research is the evaluation of the quantity of relevant documents an IR system retrieves in response to a user query. There are two important metrics used to compare information retrieval techniques, which are precision and recall. Precision is defined as the proportion of retrieved documents which are relevant, and recall as the proportion of relevant documents which have been retrieved.

When a query is given it is possible to evaluate the retrieval performance by using the values of precision and recall computed as follows [12]:

$$\text{Recall} = \frac{\text{The number of relevant documents retrieved}}{\text{The total number of relevant documents}}$$

For example, if a query is "cat" and then 100 documents returned, of which 60 documents relevant to in the database, 30 of which are about cat. The formula to calculate recall would be:

$$\text{Recall} = \frac{30}{60}$$

$$= 50\ \%$$

Trying to accomplish a high recall rate on the internet is difficult, due to the enormous volume of information that must be searched. In an ideal world, recall is 100%. However, this is impossible to achieve since the system attempts to maximize both recall and precision simultaneously.

$$\text{Precision} = \frac{\text{The number of relevant documents retrieved}}{\text{The total number of documents retrieved}}$$

For example, if a query is "cat" and then 100 documents returned, of which 60 documents relevant to in the database, 40 of which are about cat. The formula to calculate recall would be:

$$\text{Precision} = \frac{40}{100}$$
$$= 40\ \%$$

Precision of 100% is also regarded as an ideal. Since the system has to program a return of just relevant document, however, the returned document might not match the seeker's query. Simultaneously, the search system attempts to maximize both precision and recall. Precision is impossible to calculate for the web, since it is impossible to determine the total number of relevant sets which could be retrieved.

To increase precision of retrieval on the web is more difficult than to increase recall [13] due to the following problems.

- The size of the web: there are millions of resources on the web that a user can download or examine. However, in searching for information, a typical query consists of a few common words. Then, search engine systems have to rank data in their own database and return the documents relevant to the query and present a few of the most relevant ones for the user to examine.

- The quality guarantee: source of problems with achieving high precision is the fact that there is no automatic quality guarantees on the web. As a result, a retrieval system can not rely on any data provided on most web pages.

Ideally, a perfect search is desired to attain both high precision and high recall that returns all and only relevant resources with precision and recall rate of 100%. Practically, one can have quick retrieval, or a high precision, but not both. It is always a balance between the two which is augmented by the need for a speedy search. As you search in the Internet, you will discover that some engines appear to have great precision, while others have greater recall.

2.2 Fuzzy logic concept

Fuzzy logic was developed by Lotfi A. Zadeh [14] in order to provide mathematical rules and functions that permit natural language queries. Fuzzy logic deals with those imprecise conditions on which a true/false value cannot be determined. Much of this has to do with the vagueness and ambiguity that can be found in everyday life.

For example, a question of *Is it HOT outside?* probably would lead to a variety of answers of which there is no exactly correct one. As most answers are related to individual's experience and knowledge and everyone seems to attempt a higher level of abstraction during the thought process. For this reason, fuzzy logic has been compared to the human decision making process.

Zadeh introduced a concept of a fuzzy set, of which its boundary is not sharp or precise. This concept contrasts with a classical concept, called a crisp set, of which its boundary is required to be precise. Figure 2.1 show the difference between Crisp set and Fuzzy set. Crisp set consists of two elements: white element and black element (White = 1, Black=0, the value in the interval [0,1] with 0 corresponding to no membership in the class and 1 corresponding to full membership.). Fuzzy set combines three elements: White, Black and Gray (the values are between 0 and 1)

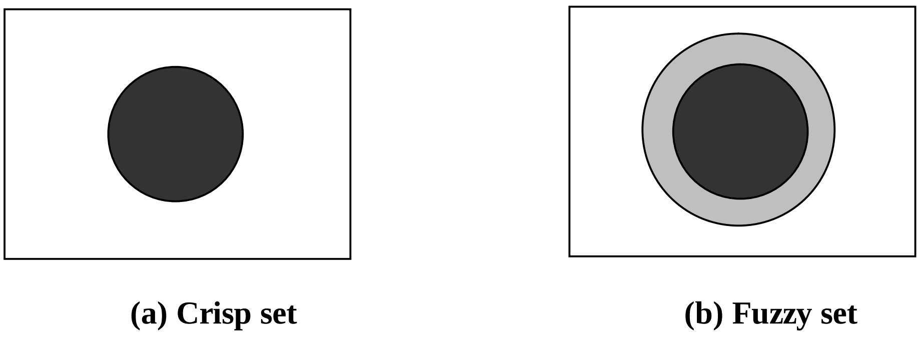
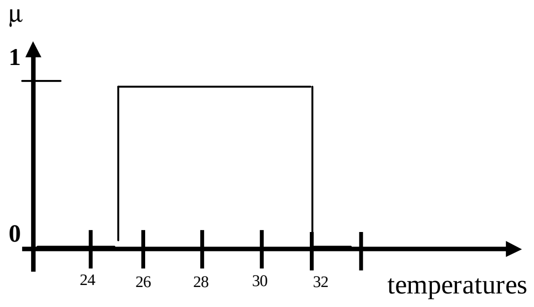


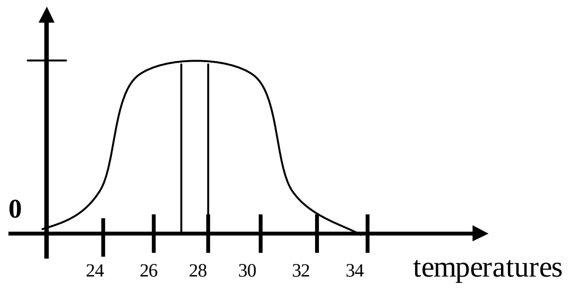
Figure 2.1 A comparison of Crisp set (a) and Fuzzy set (b)

Contrary to the classical crisp sets, the fuzzy set does not have sharp boundaries. That is, being a member of the fuzzy set is not a simple matter of being definitely true or false.

For example [14], the concept “warm” includes a wide range of temperatures. This concept may be regarded as the name of a set, WARM, which has many different temperature readings. In a given context, say a weather forecast at specific place and time is depend on a conception and specifics of the context (place, season, day, night, etc). There are several possible temperature readings, such as 25°C, 27°C, 28°C and 32°C. When asking what temperature is compatible with the concept of warm, it is likely that it would make the same judgement about the readings 27°c and 28°c, though this is not certain for every situation that can define this fuzzy set by assigning to each temperature a number between 0 and 1, which indicates the degree or grade of membership in the set. The assignment of a grade of membership in the fuzzy set of WARM to each considered temperature is called a *membership function* of this fuzzy set. An example of this function is shown in Figure 2.2



(a) Degree of temperatures with two-valued membership



(b) Degree of temperatures with multivalued membership

Figure 2.2 Membership function

Membership function - also called "fuzzy subset", a membership function is defined by a shape that defines the degree of membership for a term The most typical membership function is a triangle shape where the end point represents 0% membership and the peak is 100% membership.

Fuzzy model theory is a formal framework well suited to model vagueness. Zadeh introduced the concept of a fuzzy set, of which its boundary is not sharp or precise.

Basic definitions

The basic notions on fuzzy sets that are useful for the definition of the retrieval model. Let U be a set of objects, called the universe of discourse, and let μ denote an element of U. A fuzzy set A in the universe U is characterized by a membership function [15]:

$$\mu_A : U \longrightarrow [0,1]$$

For each $u \in$ U, $\mu_A(u)$ denotes the grade of membership of u in the fuzzy set A. When $\mu_A(u) = 1$, u is a full member of A; when $\mu_A(u) = 0$, u does not belong to A at all; and other values represent intermediate situations. Following the current notation, the fuzzy set A is represented as:

$$A = \{\mu_A(u)/ u \mid u \in U\}$$

Fuzzy information retrieval is to adopt a thesaurus (which defines term relationships). The basic idea is to expand the set of index terms in the query with related terms (obtained from thesaurus) such that additional relevant documents can be retrieved by the user query. A thesaurus can also be used to model the information retrieval problem in terms of fuzzy sets as follows.

A thesaurus can be constructed by defining a *term-term correlation matrix c* (called keyword connection matrix) whose rows and columns are associated with the document collection. In this matrix c, a normalized correlation factor $c_{i,l}$ between two term k_i and k_l can be defined by:

$$c_{i,l} = \frac{n_{i,l}}{n_i + n_l - n_{i,l}}$$

where:

n_i is the number of documents which contain the term k_i

n_l is the number of documents which contain the term k_l

$n_{i,l}$ is the number of documents which contain the term k_i and k_l

The term correlation matrix c can be used to define a fuzzy set to be associated with each index term $k_{i.}$ In this fuzzy set, a document d_j has a degree of membership $m_{i,j}$ computed as

$$m_{i,j} = 1 - \underset{kl \in dj}{P} \ (1 - c_{i,l})$$

Which computes an algebraic product overall terms in the document $d_{j.}$ A document d_j belongs to the fuzzy set associated with the term $k_{i.}$ if its own terms are related to k_i. Whenever there is at least one index term k_l of d_j which is strongly related to the index k_i (i.e., $c_{i,l} \sim 1$), then $m_{i,j} \sim 1$ and the index k_i is a good fuzzy index for the document d_j . In the case when all index terms of d_j are only loosely related to $k_{i,}$ the index $k_{i,}$ is not a good fuzzy index for d_j (i.e., $m_{i,l} \sim 0$).

Most of the existing information retrieval systems are based on the traditional Boolean logic model. The information retrieval systems based on the Boolean logic model all assume that the document and the users' queries could be represented by index terms exactly. This makes these systems restricted in practical usage especially in the environment where the information has uncertainly or fuzziness. In order to improve the drawbacks of the traditional Boolean logic model, some models like the probability model, the fuzzy set model, and the vector space model have been presented. Since the fuzzy set model can represent the inexact and uncertain knowledge of human beings, many researchers are attempting to use the fuzzy set model in the designing of fuzzy information retrieval systems. Moreover, many information retrieval techniques have been presented as follows:

Application of fuzzy technologies [16]. A concept representation model (FIS-CRM), which is able to retrieve groups of web pages that are related and improve the quality of the search results. The FIS-CRM is based on two types of fuzzy interrelations between words, synonymy and generality, obtained from a fuzzy dictionary of synonyms and a fuzzy ontology respectively. The model has been put into the metasearcher FISS (Fuzzy Interrelations and Synonymy based Searcher).

An information retrieval uses fuzzy concept networks for knowledge representation is given in [17]. This fuzzy concept network consists of nodes and links. Each node in a fuzzy concept network represents a document or a concept. Each link in a fuzzy concept network connects two concepts and is labeled with a real value between 0 and 1, which represents the relevant degree between two concepts. By means of fuzzy inference through fuzzy concept networks, the information retrieval systems are developed. Since the fuzzy inference through the fuzzy concept network is time consuming.

A fuzzy ontology-based abstract search engine [18]. Many studies show that this search engine can help users find information on the internet more effectively. In addition, a query refinement is one of the most important features of the system. In response to a user query, the system displays search result based on a standard keyword-based retrieval. The system uses a fuzzy ontology of term associations to support the feature. The ontology is automatically built in two stages using information obtained from the system’s collection.

Fuzzy conceptual-based search engine using conceptual semantic indexing [19]. The measure or techniques exist for locating exact matches, finding relevant partial matches might be a problem for internet applications such as a search engine. A requested piece of information becoming a bottleneck, is usually not a problem for small domains, but can be for large repositories such as the World Wide Web. Thus, a flexible retrieval algorithm is required, allowing for imprecise specification or search. Therefore, the task is user-defined queries to retrieve useful information according to certain measures such as fuzzy logic is required.

2.3 Search engine

2.3.1 Search engine architecture

A search engine [20] is the most popular tool for searching for information on the internet. Web users have to enter terms (keywords) in a search box that is provided by the search engine. The basic architecture of a search engine is shown in Figure 2.3.

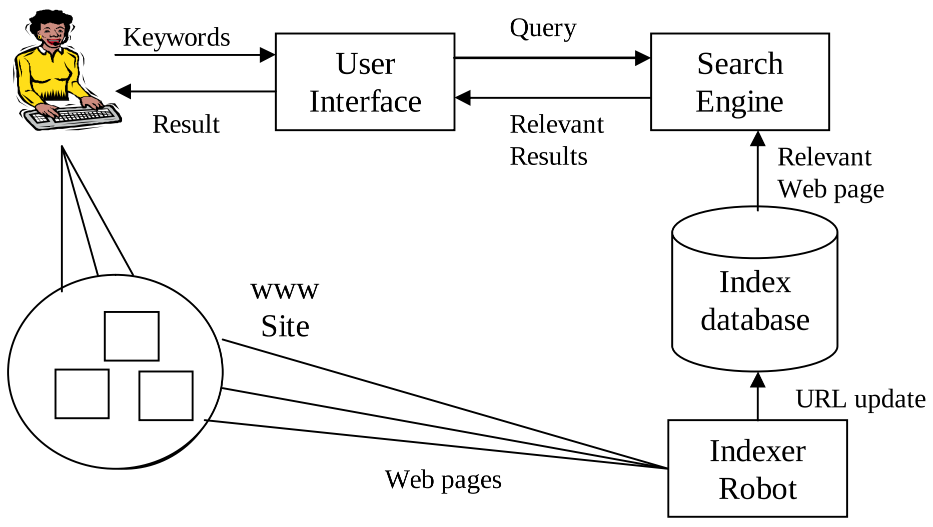


Figure 2.3 Search Engine Architecture

A search engine consists of two parts that are a back-end process(server side) and front-end process(client side) to facilitate the user to type the search terms. The interface between the client and server side consists of matching the query with the entries in the database and retrieving the information to web users.

On the server side, there are two modules:

- *Indexer robot* (*Crawler/Robot/Spider/Wanderer/Walker)* is a program that traverses the web and sends new or updated pages to a main server where they are indexed. The indexer robot runs as a local system and sends requests to a remote web server.
- *Index database* creates indices for collected web pages in the database after the indexer robot completed its tasks.

On the client side:

The search engine also provides a query interface, a box where one or more words can be typed, web users put terms (keywords) that they want to search for in the query interface. The client side (front-end process) passes the query to server side (back-end process), then the results are returned to the web users.

2.3.2 Types of the search engines

The search engines can be classified by the methods that they use to create their databases and index. The following are the types of methods available today [12]:

- Search bots: the first-generation search engines such as Alta Vista or Excite are the earliest services to help web users to find and choose information resources on the web according to their needs. The usual mode of operation of these search engines is a software agent called robot or crawler periodically downloads web pages from a portion of the web by following hyperlinks from already known pages. These pages are analyzed and certain information about them is recorded in the database of the search engine. The recorded information includes some data about the page such as its address, date of modification, language, size, as well as the information about the words composing each page and their positions on the page. Sometimes some hyper-linkage information such as lists of links pointing to each page is also recorded.

- Directories: This search method depends on humans who created the database. The common approach of all major web directories is that web resources are categorized into a hierarchical directory and are given short descriptions (and titles) by editors working for the directory. A search will look for matches only in the descriptions submitted. Yahoo is an example of this type.

- Hybrid search engines: This search method includes the capabilities of both search bots and directories, but it is not guarantee that the web pages which the owner added to the system will be included to their database. The editors of the search engines only choose to add those that look appealing. They also use automatic web crawling to create some parts of their entire database.

2.3.3 Collect data from HTML META tags

Search engines (*crawler* component) can read every linked web page that they find at every site. From this, they could build up a huge keyword database of each web page. It is thus necessary for the web page authors to identify important keywords so that the search engines can prioritize these in their indexes.

HTML is a markup language that allows to organize the presentation of a document content by means of special tags inserted in the text of the document by the author and interpreted by the HTML browsers. Most HTML elements are identified in a document by a start tag, which gives the element a name and its attributes, followed by the textual content and the end tag.

HTML META tags

- HTML META tags provide information about the web page.
- Each META tags consists of a name and content.
- There are two basic HTML META tags: description and keywords.

```
<head>
<title> My Thesis </title>
<meta name="description"
          content="Everything you wanted
                    to know about search engine,
                    and metasearch engine.">
<meta name="keywords"
          content="information retrieval, search engine,
                    fuzzy logic, content-base clustering
                    method">
</head>
```

Figure 2.4 HTML META tags

- The content attribute of the description tag specifies a paragraph of text that summarises the content of the web page.
- The content attribute of the keywords tag specifies a number of keywords that describe the web page.

META tags with attribute are classified into two categories [21].

- **name =** describing the whole HTML document.
- **http-equiv =** supplying specific fields of the HTTP headers, which are used by the browser when the page is loaded.

```
<meta http-equiv="Content-Language" content="en-GB">
<meta http-equiv="expires"
            content="Wed, 26 Feb 1997 08:21:57 GMT">
<meta name="robots" content="noindex,follow">
```

Figure 2.5 HTML META tags is classified into two categories

- **noindex** prevents anything on the page from being indexed.

- **nofollow** prevents the crawler from following the links on the page and indexing the linked pages.
- **noimageindex** prevents the images on the page from being indexed but the text on the page can still be indexed.

2.3.4 How often does a page change

Many papers investigate methods for building an effective crawler and how to improve quality of the collection. Such study shows us how to estimate the change frequency of a web page [22], which can analyze times for a web page to change. For example, if a page kept in the computer for 70 days, and that page changed 7 times in a period, it is estimated that the *average change interval* of the page of 70 days/7 = 10 days.

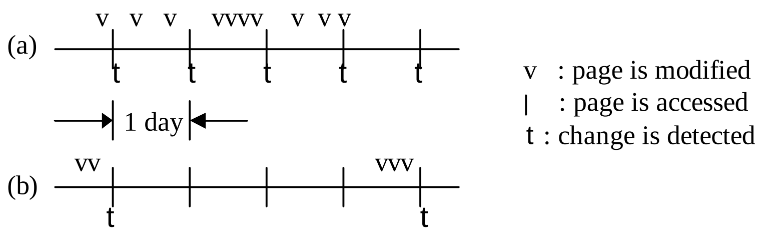


Figure 2.6 The cases when the estimated change interval is lower than the real value

Figure 2.6(a) shows the estimated change interval is one day, although a page changes more often (a page changes several times a day and then remains unchanged). This is because it can detect at most one change per day. Figure 2.6(b) interprets the estimation as the interval between the *batches of changes*, which might be more meaningful than the average interval of change. The estimated interval might be much longer than the true value.

Note that it is not easy to estimate the *average* change interval over all web pages, because the system does not know exactly how often a page changes and conducted the experiment for a limited period. The system knows how often a page changes if its change interval is longer than one day and shorter than 4 months. When its change interval is out of this range, as a crude approximation, it assumes that the pages in the first change every day and another pages change every year the overall average change interval of a web page is about 4 months [22].

In summary, the system does not know how web pages change, because a page changes rapidly, and the actual rates vary dramatically from site to site. Thus, a good search engine that is able to effectively track all these changes will be able to provide much better data than one that is not sensitive to changing data.

2.3.5 Evaluation of search engines [23]

The quality of a search engine may be determined from a number of parameters as follows:

1. The compound of its web index: coverage, update frequency, portions of web pages indexed.
2. Search capabilities: query notations, etc.
3. Retrieval performance: precision and response time.
4. Output options: for matting, fields provided, sorting capabilities.
5. User effort required: documentation, learning curve and user interface.

2.4 Metasearch engine

Although world wide web has been expanding very fast, the coverage of the web by each search engine has been decreasing despite the effort to index more web pages (from as much as 30 % to less than 10% in less than two years [1]). Search engines which have larger and higher percentage of their indexed information become obsolete. The alternative approach for providing the search capability for the entire web is by setting up a metasearch engine.

The metasearch engine is just an interface of many local search engines. It does not maintain its own index on web pages but can provide uniform access to local search engine. When a metasearch engine receives a user query, it passes the query to the appropriate local search engine and collects the results from its local search engines.

Being just an interface, it does not maintain its own index on documents. However, a sophisticated metasearch engine may maintain information on the contents of each of its underlying search engines (to be called the representative of the search engine/database) in order to provide better service. The basic architecture of a metasearch engine is shown in Figure 2.7.

2.4.1 Metasearch engine architecture

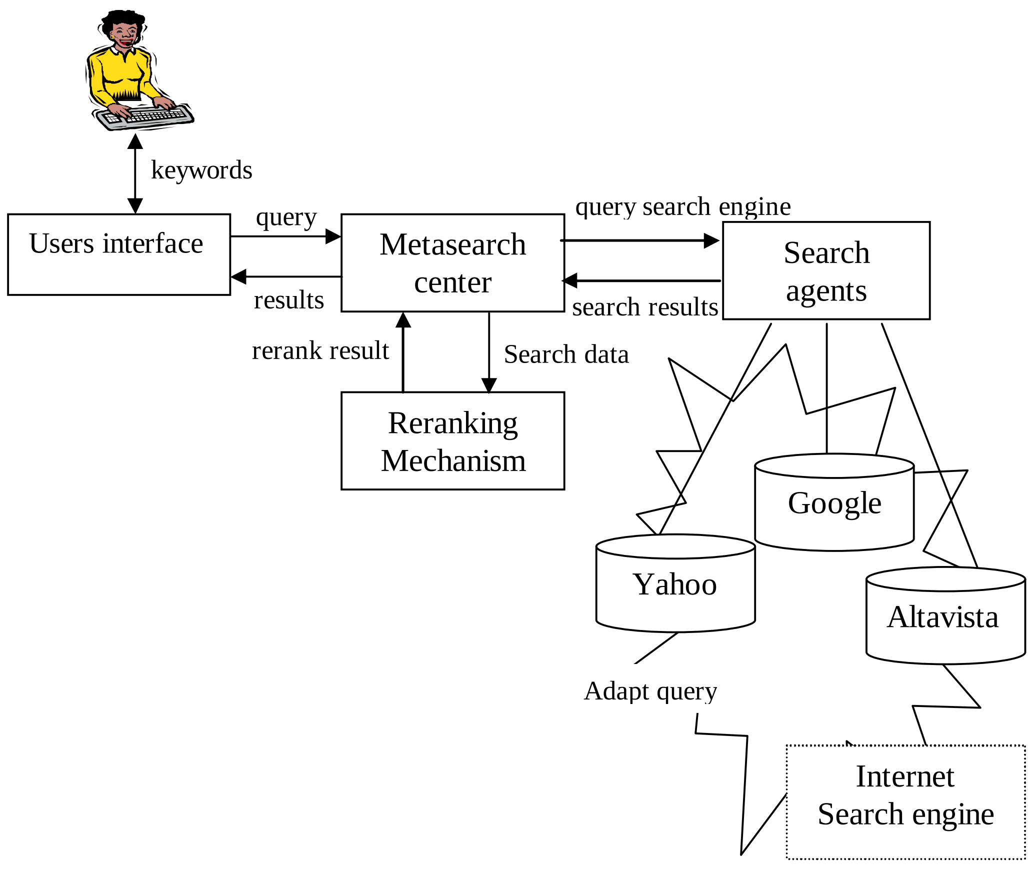


Figure 2.7 Metasearch Engine Architecture

- Users interface: the metasearch engine provide a query interface, a box where one or more words can be typed, web users put terms (keywords) that they want to search in the query box. It will passes query to metasearch center, then it returns the results to web users.

- Metasearch center: this component gets user's query from user interface. Then, it passes the user's query and target search engine names to the search agent component. When the search agents send the results back, the metasearch engine will redirect data to reranking mechanism component. Finally, the metasearch engine will return reranked results to the users interface.

- Search agents: they provide interaction of appropriate search engines such as Google, Yahoo, etc. and adapt the user's query to fit each search engine format. After that, the search agents will pass the results from the target search engine to the metasearch center.

- Reranking mechanism: this component uses own ranking technique to rerank the data from metasearch center to be uniform format and send rerank results back to the metasearch center.

2.4.2 Advantages of a metasearch engine

The main advantages of an efficient metasearch engine are:

1. Provide up-to-date information:
   A metasearch engine can maintain lists of references to resources and update information on the web by receiving results from target search engines. Each search engine will return top list results and most relevant information to the metasearch engine. Search coverage results with up-to-date information will go back to users.

2. Capability to find relevant documents:

   The metasearch engine aims to achieve this objective because there is a large number of URLs and a large amount of information in the web. Many metasearch engines must be designed to incorporate many algorithms in order to obtain all the relevant information.

3. Reduce the maintaining of an index:

   As a metasearch engine is just an interface, it does not maintain its own index on documents. When a metasearch engine receives a user‘s query, it first passes the query to appropriate local search engines, and collects the results from its local search engine, then present the results in one uniform format.

4. Provide an unified result:

   A metasearch engine has the ability to combine the results from many sources into an unified result.

5. Enhancing time saving:

   The limitation of index of search engine has caused a lot of time consuming process when web users search for the information they need. The metasearch resolves this problem by collecting the results from many local search engines to get relevant information, rather than using several single search engine to do so.

2.4.3 Estimating the usefulness of search engines

When the search engines are requested by a metasearch engine, inefficient results may occur. Then unnecessary network traffic will be created when the query is sent to useless search engines. In addition, local resources will be wasted when useless databases are searched. A better approach is to first identify those search engines that are most likely to provide useful results to a given query and then pass the query to only the identified search engines.

The usefulness of a search engine to a given query is measured by a pair of numbers; NoDoc and AvgSim. *NoDoc* is the number of documents in the database of the search engine which has similarities with the query as measured by a certain global similarity function and higher than a specified threshold, while *AvgSim* represents an average similarity of these high similarity documents. (The global similarity function may or may not be the same as the local similarity function employed by a local search engine). *NoDoc* and *AvgSim* are defined precisely below [24]:

$$NoDoc(T,q,D) = |\{d|d\,\hat{I}\ D \text{ and } sim(q,d) > T\}|$$

$$AvgSim(T,q,D) = \frac{\underline{Sd\,\hat{I}D\wedge sim(q,d) > T^{sim(q,d)}}}{NoDoc(T,q,D)}$$

T = time in searching

D = document in database

q = query

2.5 Clustering algorithm

Frequently, the results of retrieved information on the search dissatisfy users. And they have to spend much more time looking for the information that suit their desire. However, a clustering approach can solve this problem.

Document clustering has been investigated for use in a number of different areas of text mining and information retrieval. It was initially proposed for improving the precision and recall of information retrieval systems. Because clustering is often too slow for large corpora and has indifferent performance, document clustering has been used more recently in document browsing, to improve the organization and viewing of retrieval results and to accelerate nearest-neighbor search.

Common characteristics of document clustering include [25]:

- A large number of documents to be clustered.

- The number of output clusters may be large.

- Each document has a larger number of features.

Web document clustering techniques can be categorized into two types depending on their clustering time. There are *pre-retrieval clustering* and *post-retrieval clustering.*

- *Pre-retrieval clustering*: this method clusters the document before the searching process. It provides web users with the subject categories such as entertainment, computer, art, education, etc. for example Yahoo. The search engine divides the data into subject categories which help users quickly go to the specific area that they looking for.

- *Post-retrieval clustering*: this method clusters the document after the searching process. It returns the results by a keyword phrase that users have entered. For example, Metafind returns the results with domain names such as commercial site (xxx.com), education site (xxx.edu), etc.; or InferenceFind returns the results with content of the results. The content groups of the results are created by finding the con-occurrence of text appearing in titles and the description of web pages e.g. Grouper.

Besides, the clustering can classify the results with the types of cluster structure they produce, which are: h*ierarchical method* and n*onhierarchical method.*

2.5.1 Hierarchical method

This method refers to the formation of a recursive clustering of the data points: a partition of two clusters, each of which is itself hierarchically clustered. The wildly used algorithm of this method is hierarchical agglomerative clustering (HAC) algorithm. Hierarchical clustering algorithms are probably the most commonly used. These algorithms are quadratic in the number of documents and are therefore too slow for the online requirements. There are the following that use the HAC method [26]:

- *Single-linkage method*: The distance between two clusters is represented by the minimum of the distance between all possible pairs of subjects in the two clusters.

- *Complete-linkage method*: The distance between two clusters is defined as the maximum of the distances between all possible pairs of observations in the two clusters.

- *Average-linkage method*: The distance between two clusters is obtained by taking the average distance between all pairs of subjects in the two clusters.

- *Ward's method*: It forms clusters by maximizing within-clusters homogeneity. The within-group sum of squares is used as the measure of homogeneity. The Ward's method tries to minimize the total within-group or within-cluster sums of squares.

2.5.2 Nonhierarchical method [27]

Nonhierarchical method : the cluster centers or the initial partition has to be identified before the technique can proceed to cluster observations. The nonhierarchical clustering algorithms, in general, are very sensitive to the initial partition. The K-mean algorithm and other nonhierarchical clustering algorithms perform poorly when random initial partitions are used. However, their performance is very superior when the results from hierarchical methods are used to form the initial partition.

Hierarchical and nonhierarchical techniques should be viewed as complementary clustering techniques rather than competing techniques.

2.5.3 The evaluation of clustering method

In general, measuring clustering effectiveness is not a trivial issue. The standard measures such as the average distance between data points and candidate class centroids is rather abstract for the system needs. Furthermore, as already mentioned, it is not clear what distance measure should be used. In most previous work on document clustering, the performance of the clustering has been measured in terms of its effectiveness over some information retrieval system. Specifically, the clustering results are used to reorder the list of documents returned by the IR system, under the assumption that the user is able to select the clusters with the highest relevant document density [28]. There are several problems in this evaluation method as follows.

- Empirical tests have shown that users fail to choose the best cluster about 20% of the time [29].

- Generating document collections by the results obtained by an IR system. Some queries are sensitive to the specific IR system and queries being used, which may result in some unclear bias over the datasets.

- This evaluation method does not measure directly how well the inherent structure of the document corpus is revealed by the clustering procedure, but rather provides indirect estimates, through the IR system performance.

To overcome these problems, a simple solution is proposed to estimate document clustering performance by tools used for *supervised* text classification tasks. In other words, since the interest is in measuring how well the clustering process can reveal the inherent structure of a given document collection, which use a standard *labeled* text classification corpus to construct the datasets, while using the labels as clear objective knowledge reflecting the dataset inherent structure. In addition, it adopts an accuracy measure used by supervised learning algorithms to satisfy our needs. Specifically, it measures clustering performance by the *accuracy* given by the contingency table of the obtained clusters and the 'real' document categories.

2.6 Keyphrase extraction

Keyphrase is an important means of document summarization, clustering, and topic search. Keyphrase gives a high-level description of document content with an intention to make it easy for readers to decide whether it is relevant or not. But there are other applications too. Because keyphrase summarizes documents very concisely, it can be used as a low-cost measure of similarity between documents, making it possible to cluster documents into groups by measuring overlap between the keyphrases assigned. A related application is topic search: upon entering a keyphrase into a search engine, all documents with this particular keyphrase attached are returned to the user.

Many journals and books provide a list of *keywords* that call these *keyphrases*, rather than keywords. Because there are often phrases of two or more words, rather than single words. A *keyphrase list* is defined as a short list of phrases (typically five to fifteen noun phrases) that capture the main topics discussed in a given document.

Keyphrases can serve diverse goals, as they share the requirement for a short list of phrases that capture the main topics of the documents [30] as follows:

1 When they are printed on the first page of a journal article, the goal is summarization. They enable the reader to quickly determine whether the given article is in the reader's field of interest.

2 When they are printed in the cumulative index for a journal, the goal is indexing. They enable the reader to quickly find a relevant article when the reader has a specific need.

3 When a search engine form has a field labelled *keywords*, the goal is to enable the reader to make the search more precise. A search for documents that match a given query term in the *keyword* field will yield a smaller, higher quality list of hits than a search for the same term in the full text of the documents.

Several methods have been proposed for generating or extracting a summary of information from text. In the specific domain of keyphrases, there are two fundamentally different approaches [31]: *keyphrase assignment* and *keyphrase extraction*. Both use machine learning methods and require training in order to extract a set of documents with keyphrases already attached.

- Keyphrase assignment seeks to select the phrases from a controlled vocabulary that best describe a document. The training data associates a set of documents with each phrase in the vocabulary, and builds a classifier for each phrase. A new document is processed by each classifier, and assigned the keyphrase of any model that classifies it positively. The only keyphrases that can be assigned are ones that have already been seen in the training data.

- Keyphrase extraction, with an approach proposed here, does not use a controlled vocabulary, but instead chooses keyphrases from the text itself. It employs lexical and information retrieval techniques to extract phrases from the document text that are likely to characterize it. In this approach, the training data is used to tune the parameters of the extraction algorithm.

Turney [32] is the first to frame keyphrase extraction as a supervised learning problem, where all the phrases in a document are potential keyphrases, but only those that match the authors' choices are the "correct" ones. Turney devised an algorithm, called Extractor, that uses a set of heuristics and a genetic algorithm to identify the phrases that are most likely to be the authors'. Barker and Cornacchia suggested an alternative strategy by identifying noun phrases using dictionary lookup, and then consider the frequency of a given noun as a phrase head within a document, then discarding those that fall below a given threshold.

2.6.1 Result of keypharse extractor evaluation

Automatic hypertext generation by application of lexical chain [33] evaluates two algorithms for automatically extraction keyphrases from documents. The first one is NRC’ s Extractor from National Research Council of Canada [34] and KEA from New Zealand Digital Library.

Table 2.1 Some statistics for each of the five journals

Topic	**Number of articles**	**Average numbers of**	
		Keyphrase per document	**Words per keyphrase**
Information Retrieval	16	3.53	1.70
World Wide Web	11	3.33	2.15
Agents	14	3.50	2.02
Security	20	4.00	1.85
Database	9	2.78	1.80

Table 2.1 shows 70 documents from five different topics. The full text of each article is available on the web.

Table 2.2 shows the training and testing performance of the two algorithms and Table 2.3 break out the training set performance for the five different topics.

Table 2.2 Performance of the two algorithms on the document collections

Keyphrase extractions algorithm	**Training set performance 70 articles**		
	Precision	**Recall**	**F-measure**
NRC's Extractor	0.324829	0.191667	0.240415
KEA Extractor	0.453061	0.264286	0.333835

Table 2.3 Performance of the two algorithms on each of the five different topics

Keyphrase extractions algorithm	**F-measure on training set articles**				
	IR 16 articles	**WWW 11 articles**	**Agents 14 articles**	**Security 20 articles**	**Database 9 articles**
NRC's Extractor	0.203947	0.220627	0.315789	0.192915	0.317739
KEA Extractor	0.322368	0.325359	0.390977	0.310526	0.327485

The experimental results show that KEA Extractor achieves the best performance on this corpus about 40% better than the NRC's Extractor (0.333835 / 0.240415 = 0.1388576).

2.7 Comparison of search engine

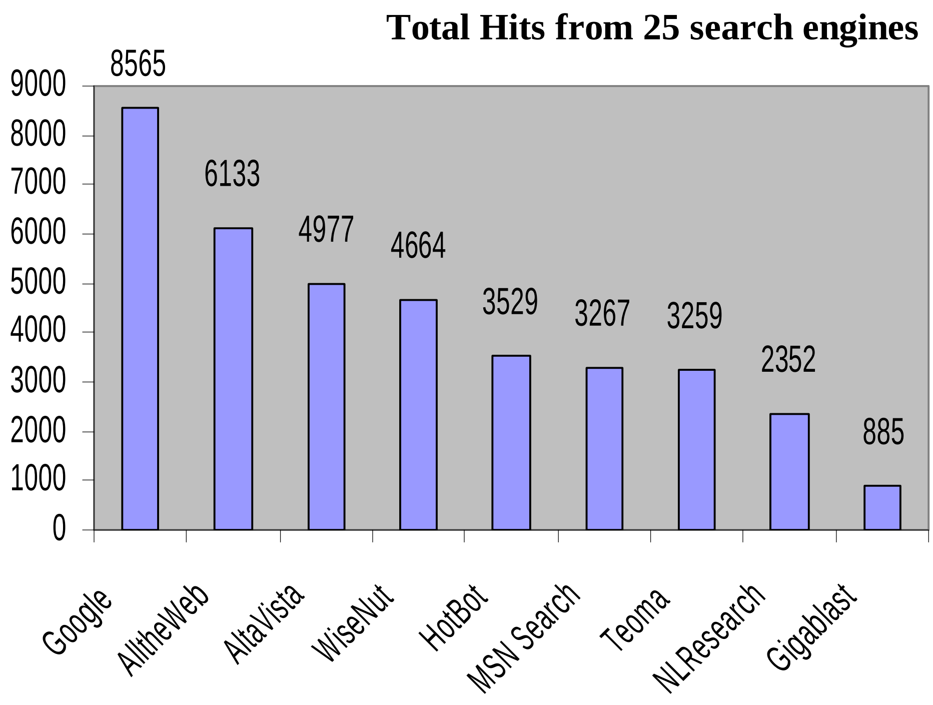


Figure 2.8 A comparison of search engines

This size showdown [35] compared nine search engines according to Figure 2.8. The analysis used 25 small single word queries. Google found more total hits than any other search engines. And ranked the first on the list of the others. AlltheWeb is ranked the second. AltaVista also had significant growth and moved up to third. WiseNut dropped to fourth and HotBot is up to fifth. Despite sharing an Inktomi source, HotBot found more total hits than MSN and included PDF files not available from MSN. For the total number of results, see Table 2.4.

Table 2.4 A comparision of total hits found by nine search engines.

Search Engine	Total hits since Dec. 2002	Total hits since March. 2002	Total hits since Aug. 2001
Google	9,732	8,371	6,567
AlltheWeb	6,757	4,388	4,969
AltaVista	5,419	3,432	3,112
WiseNut	4,664	5,009	4,587
HotBot	3,680	2,869	3,277
MSN Search	3,267	2,523	3,005
Teoma	3,259	1,839	2,219
NLResearch	2,352	3,610	3,321
Gigablast	2,352	NA	NA

2.8 Content-based clustering

Content-based clustering metasearch engine [36] develops a post-retrieval clustering call a content-based clustering (CBC) technique to help users handle the returned results of metasearch engines easier by using the contents of resultant documents for clustering. This technique will divide the results into the meaningful and accurate groups that will help users understand relationship of the results both inside a group and among the groups.

Content-based clustering aims to yield the clusters that fulfill the three basic requirements by applying (i) a feature extraction tool, (ii) a hybrid clustering of heuristic model-based of nonhierarchical clustering method and a single link of hierarchical clustering method. The feature extraction tool is used NRC’s Extractor. The heuristic model-based is used to produce an easy-to-read cluster labels and the single link method is used to merge clusters that have some latent relationship together to extend cluster coverage.

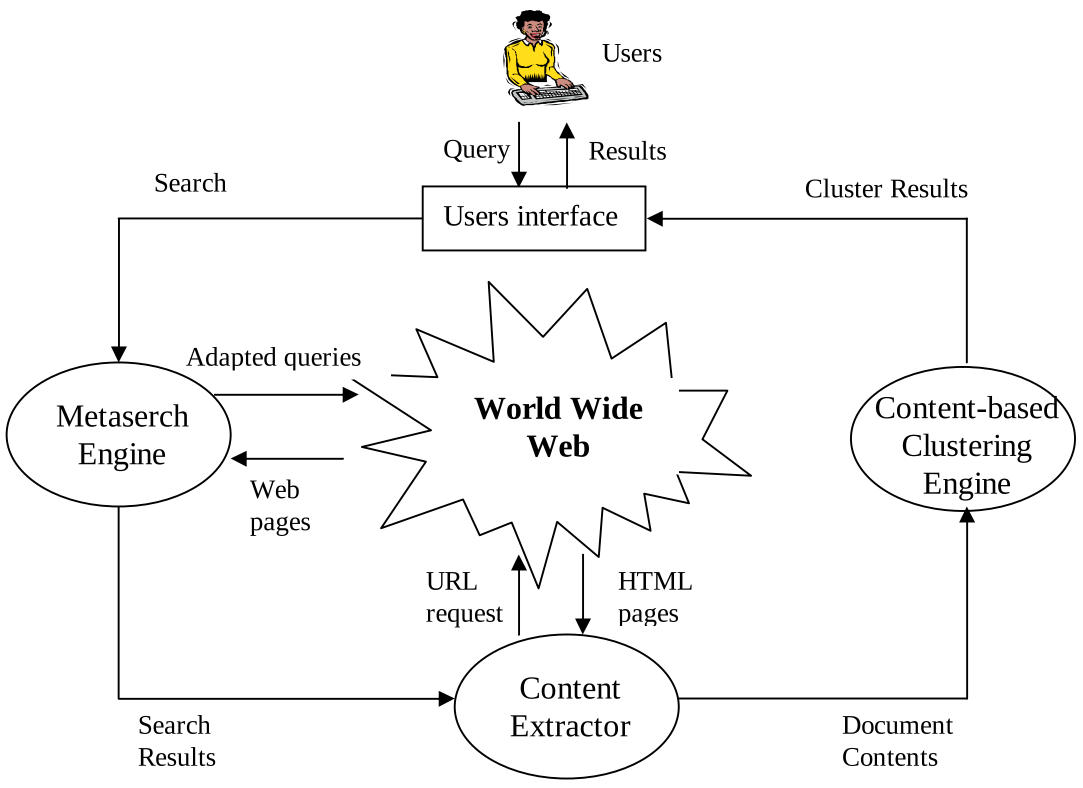


Figure 2.9 Content-based clustering metasearch engine architecture

The Content-based clustering metasearch engine (CBCMSE) architecture is shown in Figure 2.9, which consists of four submodules as follows:

- **The user interface**: this module provides communication between web users and the CBCMSE system. It passes a user's query to the metasearch engine and receives the results which are the clustered results from the content-based clustering engine.
- **The metasearch engine**: this module responsible for adapting the user's query and passing the user' s query to target search engines, which were previously registered in the system. Also, it receives the returned results from each search engine to parse and rerank them by using boost-up ordering algorithm.

- **The content extractor**: this module uses the URLs from the results of the metasearch engine to requests the HTML from each URL site. When it receives the HTMLs, it uses the NRC's Extractor to extract the contents from each HTML document and sends the contents to the content-base clustering engine.
- **The content-based clustering engine**: this module performs two important functions. First, it uses the common-phrase heuristic clustering method to find the initial clusters. Second, it merges some initial clusters that have some relationship by using the cluster merging method.

CHAPTER 3
APPROACH

This chapter describes the details of fuzzy ranking concept, content-based clustering, keyphrase extraction and boost-up ordering algorithm. It proposes explaination of all steps, as well as the examples for making the metasearch work. In fuzzy model ranking, there are four steps: using keyphrase extraction, computing *term-term correlation matrix,* calculating membership degree of terms and finding the highest ranked results. In content-based clustering concept, there are two important steps: using the common-phrase heuristic clustering method to find the initial clusters and merging some initial clusters that are relate with the cluster merging method.

Also the chapter explains about keyphrase extraction, boost-up ordering algorithm and image search technique to make the metasearch engine completed.

3.1 Fuzzy model method

The aim of an information retrieval deals with the representation, storage, organization and access to information items. The representation and organization of the information items should provide the users with easy access to the information in which they are interested. Most of the existing information retrieval systems are based on the traditional boolean logic model. The information retrieval systems based on the boolean logic model all assume that the document and the users' queries could be represented by index terms exactly.

In recent years, a branch of research in information retrieval has faced with the problem of vagueness and imprecision. This makes the systems restricted in practical usage especially in the circumstance where the information has uncertainty or fuzziness. In order to improve the drawback of the traditional boolean logic model, some models like the probability model, the fuzzy set model, and the vector space model have been presented. Since the fuzzy set model can properly represent the inexact and uncertain knowledge of human beings, many researches are devoted to the use fuzzy set model in the designing of fuzzy information retrieval systems.

Fuzzy model theory is a formal framework well suited to model vagueness. The fuzzy model according to Figure 3.1 works by:

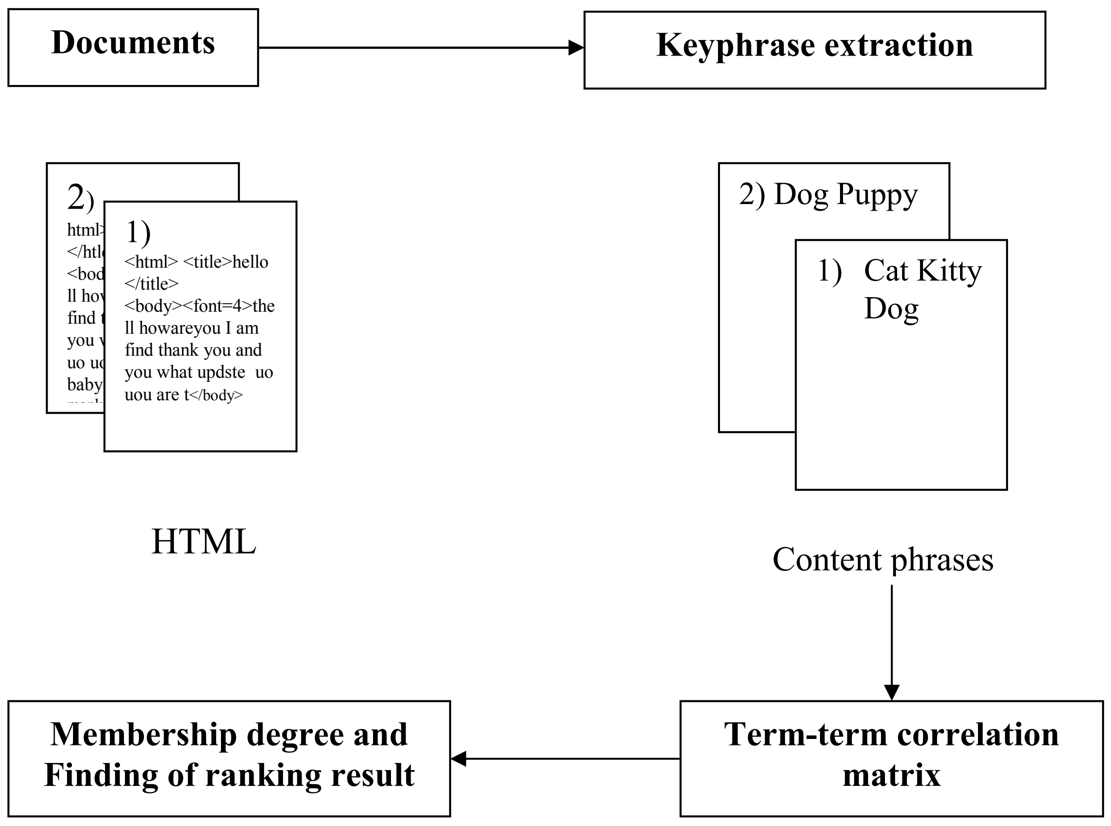
1 Using keyphrase extraction (KEA) tool to extract important keyword on content phrases in the documents.

2 Computing *term-term correlation matrix* of which its row and column are associated with the terms in the document.

3 Calculating membership degree of terms of each document.

4 Finding the highest ranked results of each document by computing weight of terms that joined together by logical connectives (*AND, OR*).

The fuzzy model can illustrate this ranking by giving an example of the computation of the relevance of the retrieved document:


Documents
Keyphrase extraction
2)
1)
<html> <title>hello </title> <body><font=4>the ll howareyou I am find thank you and you what updste uo uou are t</body>
HTML
2) Dog Puppy
1) Cat Kitty Dog
Content phrases
Membership degree and Finding of ranking result
Term-term correlation matrix

Document Number	Relationship
D1	1
D2	0.867
D3	0.673

Index 1	Index 2	Relate
Cat	Kitty	0.5
Cat	Dog	0.25
Dog	Puppy	0.5

Figure3.1 Fuzzy Model Concept

This section explains Figure 3.1 by assuming that web users want to search the following terms:

Term : *dog* Weight : *0.3*

Term : *cat* Weight : *0.5*

Operator between *dog, cat* = *OR*

Table 3.1 shows many documents that are returned from the targeted search engine and sent to keyphrase extraction process for an extraction of only important keyword.

Table 3.1 An example of index terms of each document

Document Number	Index Terms
D1	dog, cat, bird, zebra, zoo
D2	cat, kitty, fish
D3	dog, puppy, house, food
D4	ant, sugar

Next, the system finds *term-term correlation matrix c* (called keyword connection matrix) of which its rows and columns are associated with the document collection. In this matrix, a correlation factor $c_{i,l}$ between two terms k_i and k_l can be defined as follows:

$$c_{i,l} = \frac{n_{i,l}}{n_{i,+} n_l - n_{i,l}}$$

where:

n_i is the number of documents which contain the term k_i

n_l is the number of documents which contain the term k_l

$n_{i,l}$ is the number of documents which contain the term k_i and k_l

The correlation factor of each index term; *dog* and *cat* is shown in Table 3.2 below.

Table 3.2 An example of related value of each index term

Index Term 1	Index Term 2	Related Value
Cat	Dog	0.33
	Bird	0.5
	Zebra	0.5
	Zoo	0.5
	Kitty	0.5
	Fish	0.5
Dog	Cat	0.33
	Bird	0.5
	Zebra	0.5
	Zoo	0.5
	Puppy	0.5
	House	0.5
	Food	0.5

To find the membership degree of each document, this system uses the term correlation matrix *c* in Table 3.2 to define a fuzzy set associated with each index term k_i. In this fuzzy set, a document d_j has a degree of membership $\mu_{i,j}$ computed as

$$\mu_{i,j} = 1 - \prod_{kl \in dj} (1 - c_{i,l})$$

which computes an algebraic product overall terms in the document d_j. A document d_j belongs to the fuzzy set associated to the term k_i if its own terms are related to k_i. Whenever there is at least one index term k_l of d_j which is strongly related to the index k_i (i.e., $c_{i,l} \sim 1$), then $\mu_{i,j} \sim 1$ and the index k_i is a good fuzzy index for the document d_j . In the case when all index terms of d_j are only loosely related to $k_{i,}$ the index $k_{i,}$ is not good fuzzy index for d_j (i.e., $\mu_{i,l} \sim 0$).

Table 3.3 Membership degree of terms *cat, dog*

Index term	Document number	Degree of membership
Cat	D1	1
	D2	1
	D3	0.33
	D4	0
Dog	D1	1
	D2	0.33
	D3	1
	D4	0

Table 3.3 shows membership degree of term *cat* and *dog* of many documents. The values of membership degree determine whether each document is relevant to a given query or not; usually given values ranging from 0.0 to 1.0. The value 1 on documents is relevant to the query and 0 on documents that are not relevant.

Table 3.4 below show the results after calculating weight of each document by product a value of each document in Table 3.3 with a value that web users define below.

Term :	*dog*	Weight :	*0.3*
Term :	*cat*	Weight :	*0.5*

Table 3.4 Membership degree of terms *cat, dog* after calculating weight

Index term	Document number	Degree of membership
Cat	D1	0.5
	D2	0.5
	D3	0.165
	D4	0

Index term	Document number	Degree of membership
Dog	D1	0.3
	D2	0.099
	D3	0.3
	D4	0

Next, finding the ranking results of each document by computing the terms that are joined together by logical connectives (*AND, OR*). According to web users define above, operator between *dog, cat* are *OR.* This thesis uses Min and Max function [14] to find ranking result of each document.

Logic operator : AND

$\mu_{a \cap b}(x) = \mathrm{Min}((\mu_a(x), \mu_b(x)))$

Logic operator : OR

$\mu_{a \cup b}(x) = \mathrm{Max}((\mu_a(x), \mu_b(x)))$

Max function selects highest value of each document and Min function selects lowest value of each document.

Table 3.5 below shows the results after use max function and the results will order rank from highest value to lowest value. D1 is the document that the value is highest, next are D2, D3 and D4 is lowest.

Table 3.5 Value of each document

Logic operator	Document number	Value of term *cat*	Value of term *dog*	Value
Cat OR dog	D1	0.5	0.3	0.5
	D2	0.5	0.099	0.5
	D3	0.165	0.3	0.3
	D4	0	0	0

The metasearch engine is useful when users want to find relevant information how one search engine rather than using several single search engines. As the number of returned results could be significant, almost metasearch engines collect the returned results for only no more than best 10% from each search engine to avoid low precision search.

For avoid low precision search, the system will set threshold to cut off low precision search by calculating auto threshold using the equation follow [36]:

Threshold = (Min+ Mean+ Max) / 3

Min	= Minimum value
Mean	= Average value of all value
Max	= Maximum value

Threshold = (0 + 0.325+ 0.5) / 3

= 0.275

The value of each document that is lower than the value of auto threshold will be cut off. The results of fuzzy logic information retrieval are shown in Table 3.6

Table 3.6 Value ranking of each document

Document Number	Value
D1	0.5
D2	0.5
D3	0.3

Within fuzzy set theory, a document is then represented by a fuzzy set of terms with index term weights as membership values. Based on these significant degrees, the retrieval mechanism has been extended with the ability to rank document in decreasing order of relevance to the user query.

The quality of the retrieval results strongly depends on the weighting function which is adopted to automatically compute the index term weights. Generally, this weighting function is defined on the basis of term occurrences in the documents.

3.2 Content-based clustering method

The system uses fuzzy logic technique to reranking results from the targeted search engine. The ranking mechanism works well when web users can formulate a well-defined query for searches. However, a lot of users formulate very short queries (70% are single word queries[7]). So they are retrieve a large of documents to sift. Also, the problem is exacerbates when the users are not familiar with the topic that they are looking for as they are novices at performing searches.

Due to the problem above, this thesis uses content-based clustering which aims to yield the clusters that fulfill a precision of results. The clustering divides the results into several groups which may match to the areas of users‘ interest.

A *Content-based clustering(CBC)* [36] is a clustering technique which uses the contents of the documents to be clustered into several groups. The contents of the documents appeared in the dataset as the descriptors rather than using all words in the documents. The reason why the system has to use the contents of documents to be clustered is because the web documents contain too many words. So the system has to project only the important words in the documents to represent the whole documents to reduce the noise of the data. This technique will adjust the data to be more appropriate attributes.

The system can illustrate the content-based clustering process as shown in Figure3.2. Assuming that there are three HTML documents to be clustered. A circle represents cluster and the capital alphabet such as A, B, C, D are represent the content phrases. The number 1, 2, 3 and 4 represent the document's numbers.

The content-based clustering works by:

- First, using index terms in table *keyword* of database that is extracted from each document by keyphrase extraction tool.

- Next, finding the co-occurrence content phrases appeared in each document in dataset to create initial clusters.

- Finally, a cluster merging method used to merge the clusters that have a relationship.

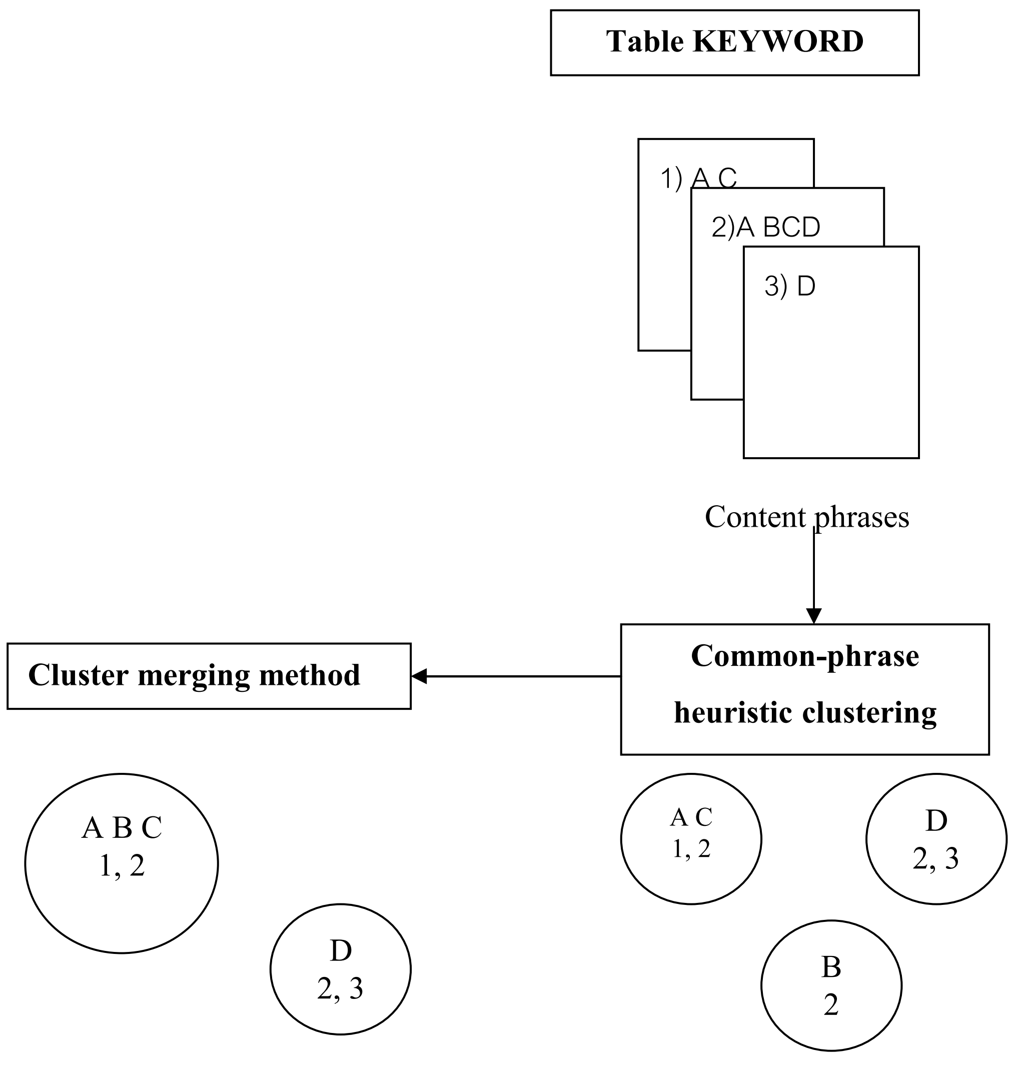


Figure3.2 Content-based clustering concept

3.2.1 Common-phrase heuristic clustering method

The content of each document projected by keyphrase extraction*(*KEA*)* with the multiple of content phrases. The common-phrase heuristic clustering method provides rules for clustering these contents. The common-phrase heuristic clustering method works by grouping the documents that have co-occurrence phrases together. However, the common phrases that represent cluster labels always look awkward, so heuristic rules are used to solve the problem.

The common-phrase heuristic clustering method works by [37]:

1 Finding all common contents appeared in the dataset.
2 Using the first rule to indicate only specific common contents to cluster representation.
3 Using the second rule to reduce awkward of those specific cluster representations.

This clustering method can be illustrated with an example of the Common-phrase heuristic clustering method. Assume the dataset according to Table 3.7:

Table 3.7 An example of documents and their contents to be clustered

Document Number	Contents
1	A B F H I
2	A B E F
3	C D F G J
4	C D F S

RULE:

RULE 1*: IF(First common contents)$\subset$(Second common contents) then (delete First common contents = Yes)*

RULE 2*: IF(First chosen common contents)overlap with (Second chosen common contents) then (First chosen common contents = (First chosen common contents – Overlapped contents)) and (Second chosen common contents = (Second chosen common contents – Overlapped contents)) and Create New Cluster (Overlapped contents)*

All of the possible common contents of these documents are:

1) A (documents: 1, 2)
2) B (documents: 1, 2)
3) C (documents: 3, 4)
4) D (documents: 3, 4)
5) F (documents: 1, 2, 3, 4)
6) A B (documents: 1, 2)
7) A F (documents: 1, 2)
8) B F (documents: 1, 2)
9) C D (documents: 3, 4)
10) C F (documents: 3, 4)
11) D F (documents: 3, 4)
12) A B F (documents: 1, 2)
13) C D F (document: 3, 4)

Obviously, there are no best common contents that are perfect representation for all users. However, the first rule proposed to the largest number of common contents is the best representation of the cluster. The assumption for this rule is that the more number of content phrases the more specific of the cluster representation. The result is shown in Table 3.8.

Table 3.8 An initial clusters

Initial Cluster	Documents
A B F	1, 2
C D F	3, 4

According to Table 3.8, it can be seen that clusters (A B F) and (C D F) have (F) in common. When the system applies the second rule, the initial cluster representations become (A B), (C D) and (F). The result is shown in Table 3.9 below:

Table 3.9 An example of initial clusters

Initial Cluster	Documents
A B	1, 2
C D	3, 4
F	1, 2, 3, 4

All processes and rules are shown in Figure 3.3 below:

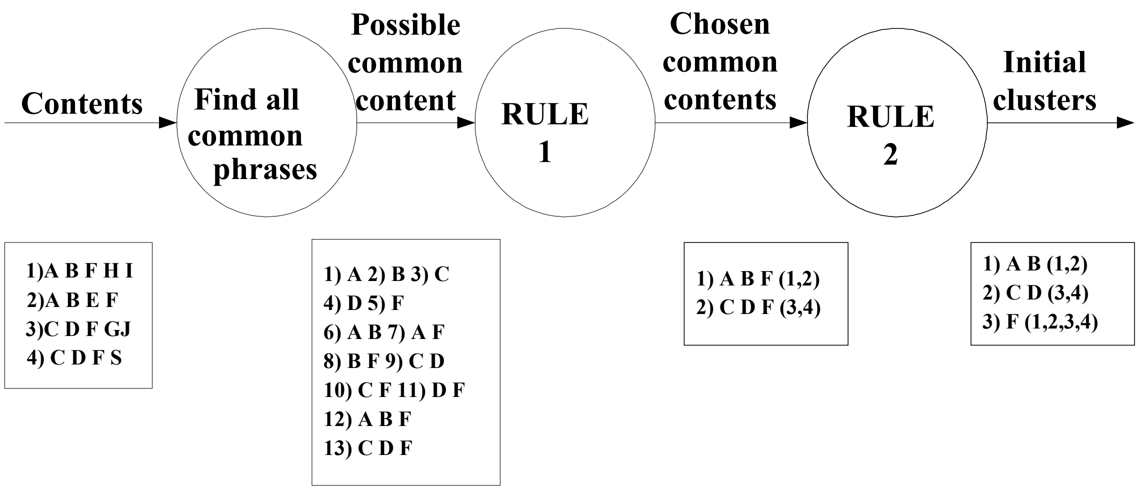


Figure 3.3 Common- phrase heuristic clustering process

3.2.2 Cluster merging method

Another process of the content-based clustering approach is cluster merging. This thesis proposes the use of single-link clustering method for handling a problem that some document are separated by each common contents appearing in the document by the common-phrase heuristic clustering method which have some latent relationship to the others rather than common contents.

The single link method is a hierarchical method, which tends to generate the clusters by similarity of the documents rather than using only common phrase, but no overlap of documents in the clusters are allowed:

The single link method works in the following forms [20]:

1. The system creates the vector space of each document in the entire dataset. The weight of each term in the vector space is calculated by TF*IDF algorithm.

2. Every pair of documents is calculated with similarity by using the cosine similarity measurement. Then, the system has the similarity metric.

3. The system uses a defined threshold to link the document together. Two documents are considered linking if their similarity value is above the threshold. After all, the system obtains a document chain.

4. The system compares the initial clusters with the document chain. If some of two or more initial clusters are in subset of document chain, those initial clusters have to merge together.

This clustering can be illustrated with an example of single link method and assumed dataset as shown in Table 3.10 below.

Table 3.10 An example of documents and their contents to be clustered

Document Number	Contents
1	A B E X Y
2	A B E Y
3	C D F Z
4	C D F W
5	E U G
6	E V I
7	F T J
8	F S R

Next, finding the co-occurrence content phrases appeared in each document in Table 3.10 to create initial clusters by *rule 1*and *rule 2* above. The result is shown in Table 3.11 below.

Table 3.11 Initial clusters of the example

Initial Cluster	Documents
A B Y	1, 2
C D	3, 4
E	1, 2, 5, 6
F	3, 4, 7, 8

Then, In single link method, the system creates a similarity metric of those documents in the dataset by using cosine similarity measurement as follows: The cosine similarity measurement [37]

This measurement is widely used for measuring in the vector space model is a cosine similarity measurement. After every document and query is translated to vector space model, a vector matching based on the cosine of the angle between two vectors can be used to compute the similarity between document d_j and query q_k as follows.

$$Similarity\ (d_j,q_k) = \sum_{t=1}^{n} (td_{ij} * tq_{ik}) / (\sqrt{\sum td^2_{ij} * tq^2_{ik}})$$

where:

td_{ij} = the weight of term i in the vector of document j

tq_{ik} = the weight of term i in the vector of query k

n = the number of all unique terms in the dataset

Similarity metric is shown in Table 3.12 below.

Table 3.12 Similarity metric of the document

Number of documents	1								
	2	0.6							
	3	0.1	0						
	4	0.2	0	0.6					
	5	0.1	0.5	0.2	0.2				
	6	0	0.7	0.3	0	0.8			
	7	0.2	0	0.8	0.5	0	0		
	8	0	0	0.5	0	0.1	0.2	0.7	
		1	2	3	4	5	6	7	8
		Number of documents							

After, the system has to define threshold to link the documents together. By calculating auto threshold using the equation as follows [36]:

Threshold = (Min(sim) + Mean(sim) + Max(sim)) / 3

Sim	= Similarity value
Min(sim)	= Minimum value in the similarity metrix
Mean(sim)	= Average value of all similarity value in the similarity metrix
Max(sim)	= Maximum value in the similarity metrix

Threshold = (0 + 0.26+ 0.8) / 3

= 0.35

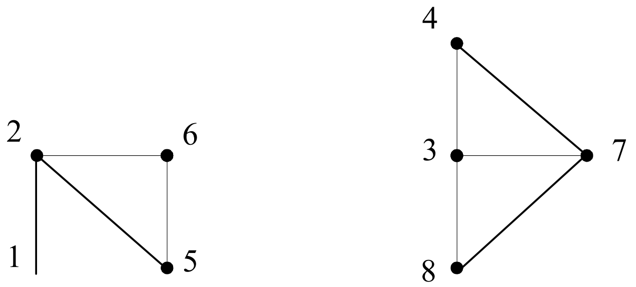


Figure 3.4 A similarity metric and chaining documents of the example

Figure 3.4 is the graphical of chaining method after the system calculates similarity metric in Table 3.12. Two documents are considered linking if their similarity value is above the threshold.

The result in Table 3.13 comes from comparing data in Table 3.11 and the chaining documents in Figure 3.4. So chaining document graphs are (1,2,5,6) and (3,4,7,8) that in Table 3.11 documents 1,2 are initial cluster (A,B,Y) and documents 2,5,6 are initial clusters (E) then documents 1,2,5,6 are chaining document labels (A,B,E,Y).

Table 3.13 Chaining document and their members

Chaining documents label	Documents
A B E Y	1, 2, 5, 6
C D F	3, 4, 7, 8

RULE 3*: IF(initial cluster members)$\subset$(chaining member) then (Merge cluster = Yes)*

This rule is used iteratively for every pair of initial clusters and chaining documents. The initial clusters and chaining documents are shown in Table 3.11 and Table 3.13 respectively. By using the third rule, the initial clusters which are (A, B, Y) and (E) have members in subset of chaining document (A, B, E, Y). These initial clusters are merged together to be (A, B, E, Y). Also, the initial clusters which are (C, D) and (F) have members in subset of chaining documents (C, D, F), so they are merged together to be(C, D, F).

Finally the merged clusters are shown in Table 3.14 below.

Table 3.14 Merged clusters and their members of the example

Merged cluster label	Documents
A B E Y	1, 2, 5, 6
C D F	3, 4, 7, 8

All processes and rules of this method are shown in Figure 3.5 below:

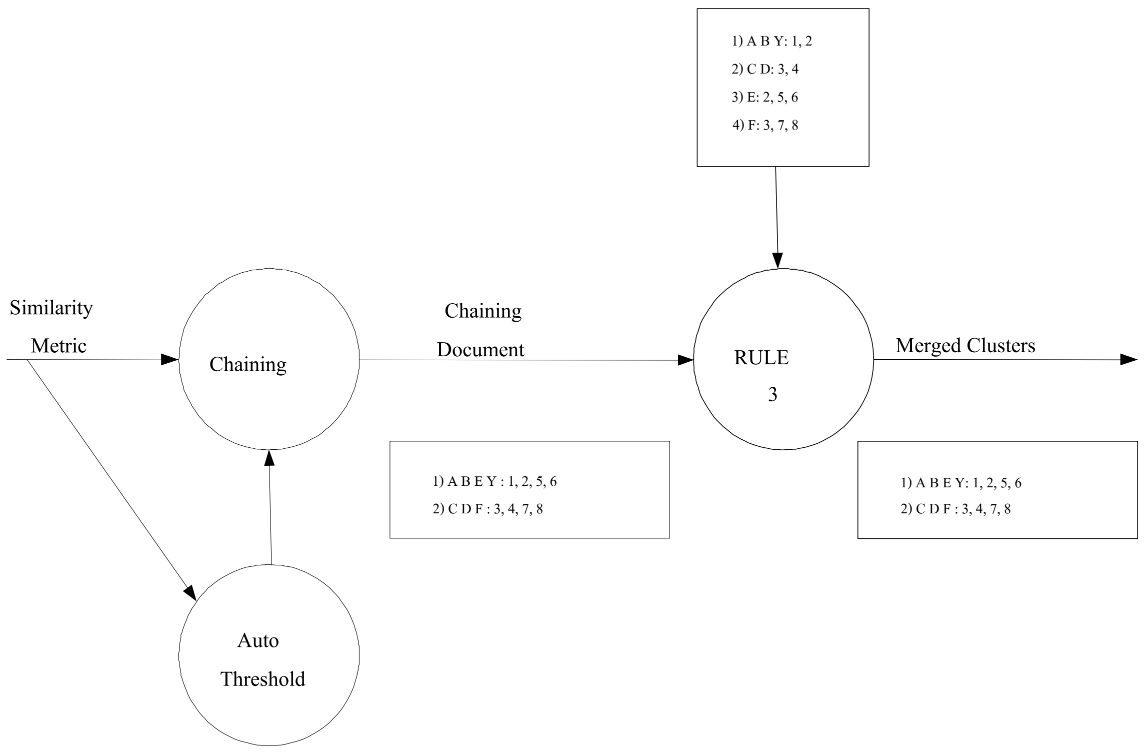


Figure 3.5 Cluster Merging Process

Since the approach adopts both model-based clustering and single link HAC, the final clusters are distinct from clusters that are produced by traditional clustering algorithms such as meaning and accuracy of the clusters. It is because the heuristic model-based is designed for producing meaningful clusters, and single link HAC is used for extending cluster coverage.

3.3 Keyphrase extraction

Keyphrases provide a brief summary of a document's contents and used in information retrieval systems as descriptions of the documents returned by a query, as the search indexes, as browsing a collection, and as a document clustering technique. The keyphrases are useful because they can be interpreted individually and independently of each other. Therefore large document collections such as digital libraries become widespread and the value of summary information increases.

Keyphrase can be automatically generated in two ways: by assigning keyphrases from a controlled vocabulary to documents, or by identifying and selecting the most descriptive phrases in the document.

In the first approach, it is called keyphrase assignment or text categorization, the controlled vocabulary ensures that similar documents are classified consistently, and allows phrases to be assigned even if they are not explicitly mentioned in their text. However, controlled vocabularies are too expensive to build and maintain, so they are not always available, and potentially useful keyphrases are ignored if they are not in the vocabulary.

In the second approach, it is called keyphrase extraction, the text of a document is analyzed and its most appropriate words and phrases are identified and associated with the document. This means that every phrase that occurs in the document is a potential keyphrase of the document, and a controlled vocabulary is not required. The keyphrases generated are less consistent, however. Automatically identifying and extracting phrases are complex tasks, but a range of techniques for identifying useful, descriptive and meaningful phrases have been suggested. Few of these conform to the definition of a keyphrase extraction algorithm. The distinction between keyphrase extraction algorithms and other phrase extraction algorithms is important because the author keyphrases provides an objective basis for evaluation, as will be discussed below.

This thesis employs keyphrase extraction to assist in fuzzy ranking and content-based clustering. And KEA – a practical automatic keyphrase extraction is considered for its best efficiency for use.

KEA is a keyphrase extraction algorithm developed by members of the New Zealand Digital Library Project. A Java implementation, released under the GNU General Public License. KEA operates in two distinct stages. First, a model is built from a set of training documents with keyphrases (usually the author of keywords, though any authoritative source may be used). Second, documents without keyphrase metadata are presented to KEA, and the model is used to identify those of their phrases that are most likely to be keyphrases. These phrases are "extracted" from the document and provided as output. The process is illustrated in Figure 3.6.

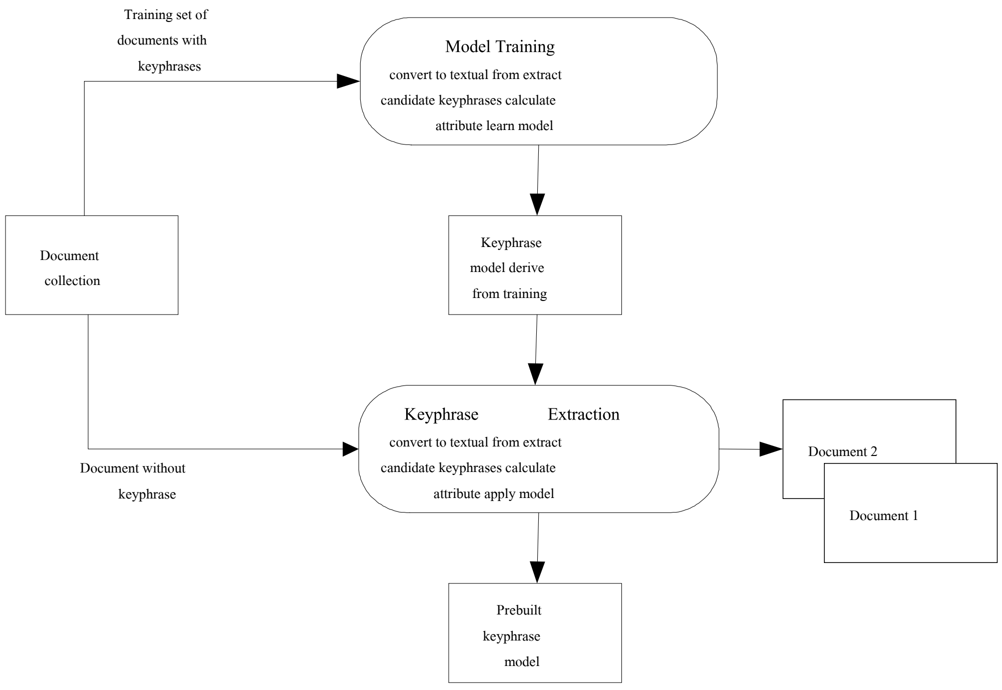

Training set of documents with keyphrases
Model Training
convert to textual from extract candidate keyphrases calculate attribute learn model
Document collection
Keyphrase model derive from training
Keyphrase Extraction
convert to textual from extract candidate keyphrases calculate attribute apply model
Document without keyphrase
Document 2
Document 1
Prebuilt keyphrase model

Figure 3.6 KEA phrase extraction process

3.3.1 The KEA algorithm

KEA's extraction algorithm has two stages:

1. Training: create a model for identifying keyphrases, using training documents where the author's keyphrases are known.

2. Extraction: choose keyphrases from a new document.

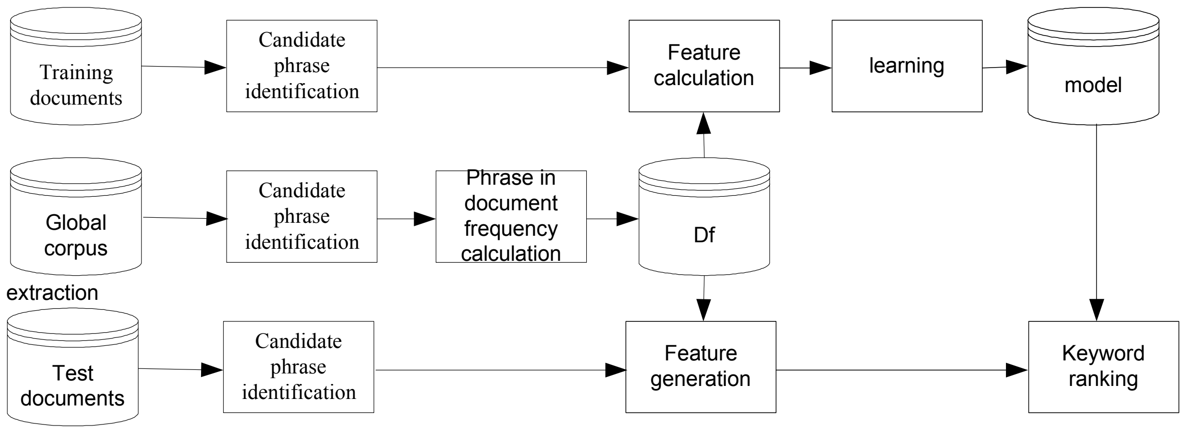

Training documents
Candidate phrase identification
Feature calculation
learning
model
Global corpus
Candidate phrase identification
Phrase in document frequency calculation
Df
extraction
Test documents
Candidate phrase identification
Feature generation
Keyword ranking

Figure 3.7 The training and extraction processes

Figure 3.7 shows the two stages of KEA algorithm, with the outline to be described in more detail.

3.3.2 Candidate phrases

KEA chooses candidate phrases in three steps as follows:

Input cleaning

The input files are filters of the text. They determine initial phrase boundaries. The input stream is split into sequences of letters, digits and internal periods and then several modifications are made as follows:

- punctuation marks, brackets, and numbers are replaced by phrase boundaries.
- apostrophes are removed.
- hyphenated words are split into two.
- remaining non-token characters are deleted, as are any tokens that do not contain letters.

Phrase identification

KEA considers all in subsequences of each line and finds which of these are suitable candidate phrases. However, there are several methods for determining suitability, such as looking for noun phrases, but the system has found that there are effective rules as the following:

- There is a limit to a certain maximum length of candidate phrases (usually three words).

- Candidate phrases cannot be proper names such as single words that only ever appear with an initial capital.
- Candidate phrases cannot begin or end with a stopword.

The stopword list contains 425 words in nine syntactic classes (conjunctions, articles, particles, prepositions, pronouns, anomalous verbs, adjectives, and adverbs). In these classes, all the words listed in an on-line dictionary are added to the list. However, for adjectives and adverbs, it introduced several subclasses, and words from the subclasses are added only if they overlap the sixty most common words in the Brown corpus [38]. In addition, it only adds frequently-occurring words from these subclasses.

Case-folding and stemming

The Lovins method [39] employed in final step of determining candidate phrases, is used to stem. This involves using the classic Lovins stemmer to discard any suffix, and repeating the process on the remaining stem until there is no further change. For example, the phrase *cut elimination* becomes *cut elim*.

Stemming and case-folding allow us to treat different variations on a phrase as the same thing. But without stemming they would be as different phrases. The stemmed versions are used to compare KEA's output with the author's keyphrases. It considers an author's specified keyphrase to have been successfully identified if, when stemmed, it is the same as a machinegenerated keyphrase, also stemmed. The phrases *cut-elimination* and *cut elimination*, and *proof nets* and *proof net*, are considered equivalent.

In addition, the system retains the unstemmed words of each phrase (in their original capitalization) for presentation to the user in case the phrase does turn out to be a keyphrase.

3.3.3 Feature calculation

TF*IDF is a measure of a phrase's frequency in a document, it is easy to implement and most widely used in vector space model.

- TF (term frequency) is a frequency of the phrase that occurs in a document.
- IDF (Inverse document frequency) is the number of other documents in which the phrase occurs.

This feature compares the frequency of a phrase used in a particular document with the frequency of that phrase in general use. General phrase is represented by *document frequency* —the number of documents containing the phrase in some large corpus. KEA builds a document frequency file for this purpose using a corpus of about 100 documents. Stemmed candidate phrases are generated from all documents in this corpus using the method described above.

TF*IDF = $tf_{ik} * \log(N/n_k)$

where:

1. tf_{ik} : frequency of term T_k in document D_i.
2. N : total number of documents in the collection C.
3. n_k : the number of documents in C that contain T_k..

First occurrence

The second feature is the first occurrence, it is calculated as the number of words that precedes the phrase's first appearance, divided by the number of words in the document. The result is a number between 0 and 1 that represents the amount of document preceding the phrase's first appearance.

Discretization

Discretization method is accomplished by using the supervised discretization method described in [40]. During the training process, a discretization table for each feature is derived from the training data. This table gives a set of numeric ranges for each feature, and values are replaced by the range into which the value falls.

3.3.4 Training: building the model

The training stage uses the Naive Bayes technique [41], because it is simple and yields good results. The scheme then generates a model that predicts the class using the values of the other two features. A set of training documents for which the author's keyphrases are known. For each training document, candidate phrases are identified and their feature values are calculated as described above. To reduce the size of the training set, it discards any phrase that occurs only once in the document and marks phrase as a keyphrase or a non-keyphrase.

3.3.5 Extraction of new keyphrases

The difference between keyphrase extraction algorithms and other phrase extraction algorithms is that the KEA attempts to extract the same phrases that the author would choose. A consequence of this intention is that there is an objective way to measure performance: if an extracted keyphrase is the same as an author keyphrase, then the algorithm succeeds; if not, then it failes. In selecting keyphrases from a new document, KEA determines candidate phrases and feature values, and then applies the model built during training. The model determines the overall probability that each candidate is a keyphrase, and then a post-processing operation selects the best set of keyphrases.

When the Naive Bayes model is used on a candidate phrase with feature values *t* (for TF*IDF) and *d* (for distance), two quantities are computed:

$$\mathrm{P}[yes] = \frac{Y}{Y+N}\, p_{TF*IDF}[t \mid yes]\, \mathrm{P}distance[d \mid yes]$$

where

- *Y* is the number of positive instances in the training files

- *N* is the number of negative instances that candidate phrases are not keyphrases.

The overall probability that the candidate phrase is a keyphrase can then be calculated:

$p = \mathrm{P}[yes] / (\mathrm{P}[yes]+\mathrm{P}[no])$

Candidate phrases are ranked according to this value, and two post-process steps are carried. First, TF•IDF (in its pre-discretized form) is used as a tie-breaker if two phrases have equal probability (common because of the discretization). Second, it removes from the list any phrase that is a subphrase of a higher-ranking phrase. From the remaining ranked list, the first *r* phrases are returned, where *r* is the number of keyphrases requested.

3.4 Boost-up ordering algorithm

This thesis is intended to develop the metasearch engine using fuzzy logic technique to retrieve information to satisfy web users that they can define weighting of keywords by themselves. Also, the system uses KEA *(*Keyphrase Extraction) to extract a brief summary of a document's contents. KEA assists in English term searching. However, this tool does not support THAI term query.

The boost-up ordering algorithm is a ranking technique which can find the highest rank order of the identical results. It deletes the duplicate link that is returned from the targeted search engine, as well as preserving the highest rank result.

For example, if a result shows document *A* ranked first at Sanook, third at Thaiseek, the rank of the document *A* will be boosted up to only Sanook rank in one uniform list. If there is no duplicated link returned by the targeted search engine, then the final result will be ordered in the highest rank from search engine. For example, if the documents from Sanook are A, B, C and the documents from Thaiseek are D, E, F. The result which pass the boost-up ordering algorithm are A, D, B, E, C, F.

This method can be illustrated with an example of the boost-up ordering approach as shown in Figure 3.8 below:

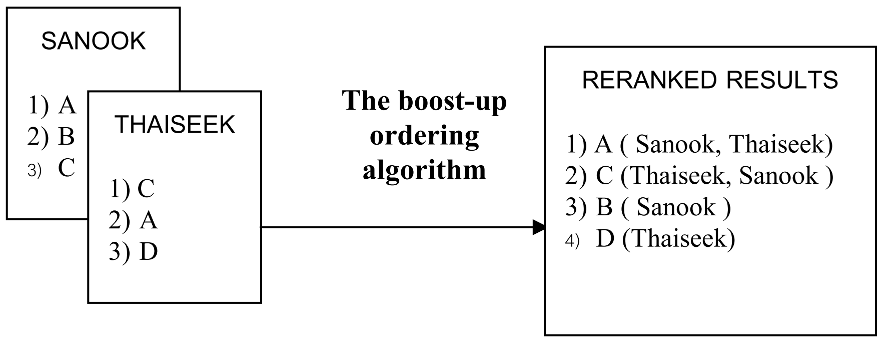


Figure 3.8 Boost-up ordering algorithm

CHAPTER 4
IMPLEMENTATION OF MEUFLACC

This chapter describes the detailed implementation of MEUFLACC model and provides an environment test to evaluate its performance. Structure chart of the metasearch engine using fuzzy logic and content-based clustering will show all important module of the system. Also this part explains all steps of the system in data flow diagram (DFD) and describes working process in flow chart to explain how the prototype works. Data dictionary is a method that shows all input data and output data and tells everything of each module. Finally, the chapter shows an implementation of user interface.

4.1 Environment test and tools

This thesis uses a notebook computer running under Windows XP Professional to evaluate the metasearch engine using fuzzy logic and content-based clustering. The specifications of the testing are provided in Table 4.1. Developing tools consist of PHP 4.2.0 programming language for coding the prototype, Apache 1.3.24 to be web server. The system uses Mysql as the relational database because it is flexible and contains all functions that are needed. This section shows developing tools in Table 4.2 below.

Table 4.1 Hardware environment.

CPU	Pentium IV, 2.0 GHz
RAM	256 MB
Hard disk	40 GB

Table 4.2 Software environment.

Compiler	PHP 4.2.0
Web server	Apache 1.3.24
DBMS	Mysql
Operating system	Microsoft windows xp professional

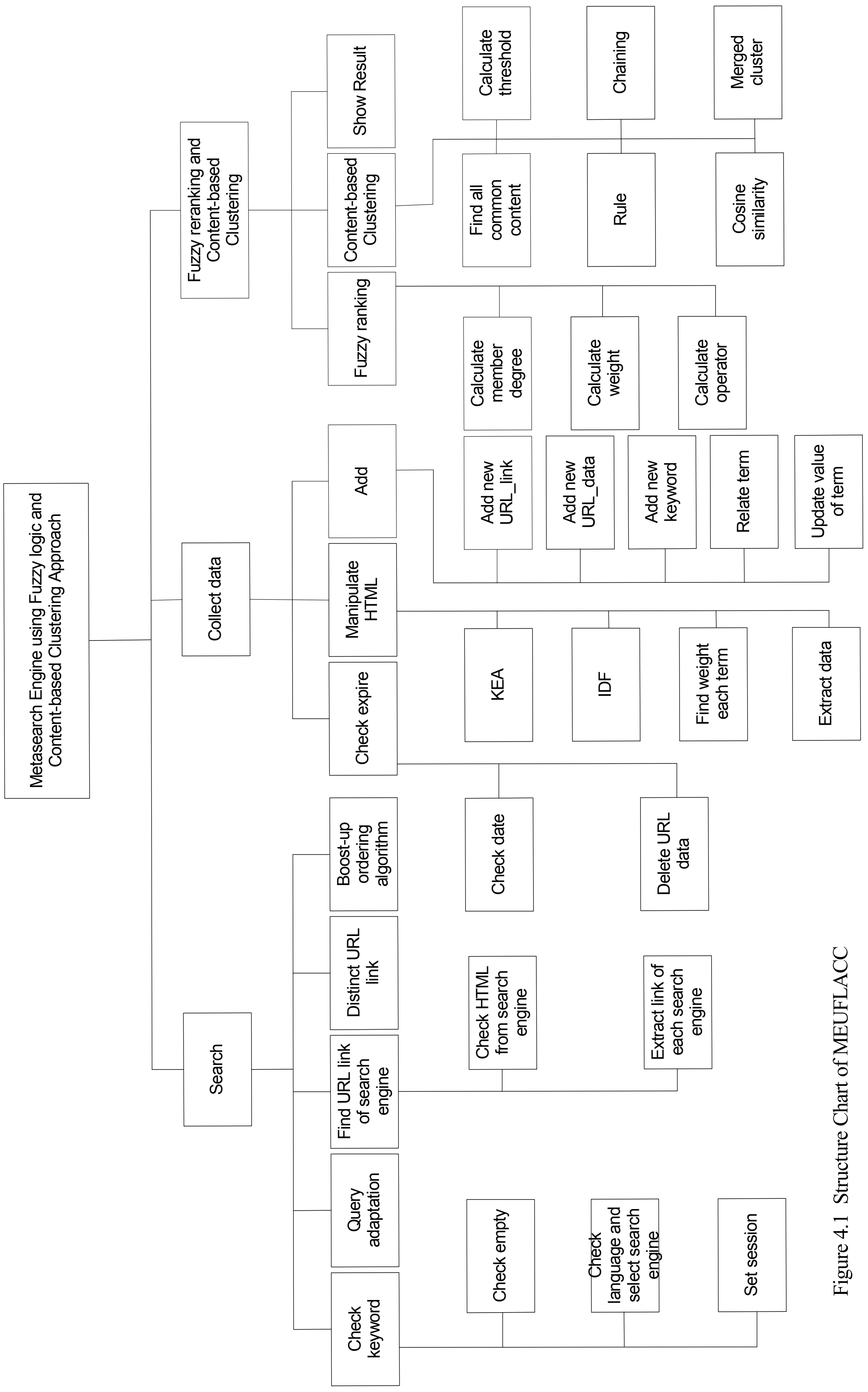


Figure 4.1 Structure Chart of MEUFLACC

4.2 Structure chart of the system

The system, illustrated in Figure 4.1, consists of three subsystems as follows:

- **Search system** is the subsystem that passes the query to the targeted search engines and extracts URL link

- **Collect data system** is the subsystem that extracts HTML from webpage and adds new data to the database.

- **Fuzzy reranking and content-based clustering** is the subsystem that reranks results from the targeted search engine by using fuzzy logic and content-based clustering

Search system has the following functions:

- Check keyword is used for checking the query from web users and examine the query that is in database or not.
- Query adaptation is used for adapting a query from web users to appropriate targeted search engine.
- Find URL link of search engine is for extracting URL link of results that return from target search engine.
- Distinct URL link is used for making URL link to be unique.
- Boost-up ordering algorithm is used for reranking results from THAI search engine.

Collect data system has the following functions:

- Check expire is a module for checking expiry date of URL link in database.
- Manipulate HTML is a module for extract important data from internet host.

- Add is a function that adds all information to database.

Search system has the following functions:

- Fuzzy ranking is used for reranking results from target search engine by fuzzy logic method.
- Content-based clustering is a module for helping a novice web users in searching information that satisfy their' s want.
- Show result is a function that shows the result from fuzzy ranking, boost-up ordering algorithm and content-based clustering method.

Check keyword has the following subfunctions:

- Check empty is a function that checks the query term from web users, if it is empty, the system will show message and wait until users insert new query term.
- Check language and select search engine is a function that checks the query term, it is THAI or English language to select appropriate search engine.
- Set session is a function that starts session for multi-users query.

Find URL link of search engine has the following subfunctions:

- Check HTML from search engine is a function that checks HTML page from targeted search engine, if search engine does not return the results, the system will show message.
- Extract link of each search engine is a function that extracts URL link from results that returned from targeted search engine.

Check expire has the following subfunctions:

- Check date is a function that examines expiry date of URL link in database.
- Delete URL date is a function that deletes all data in database, URL link has expired.

Manipulate HTML has the following subfunctions:

- KEA is a function that extracts important word of webpage.
- IDF is a function that calculates value of IDF for use in content-based clustering method.
- Find weight each term is a function that calculates weight of each term.
- Extract data is a function that extracts URL, title and description of each webpage.

Add has the following subfunctions:

- Add new URL_link is a function that adds new URL link in database.
- Add new URL_data is a function that adds new data in database.
- Add new keyword is a function that adds new index term in database.
- Relate term is a function that calculates term-term correlation.
- Update value of term is a function that updates new value of term in database.

Fuzzy ranking has the following subfunctions:

- Calculate member degree is a function that calculates member degree of each document.
- Calculate weight is a function that calculates weight of each document.

- Calculate operator is a function that calculates Boolean operator of each document.

Content-based clustering has the following subfunctions:

- Find all common content is a function that finds all common content of all document.
- Rule is a function that checks rule of all document.
- Cosine similarity is a function that calculates cosine similarity of each document.

Show result has the subfunctions as follows:

- Calculate threshold is a function that calculates threshold.
- Chaining is a function that finds chaining.
- Merged cluster is a function that merges initial clusters and chaining document.

4.3 Flowchart of the system

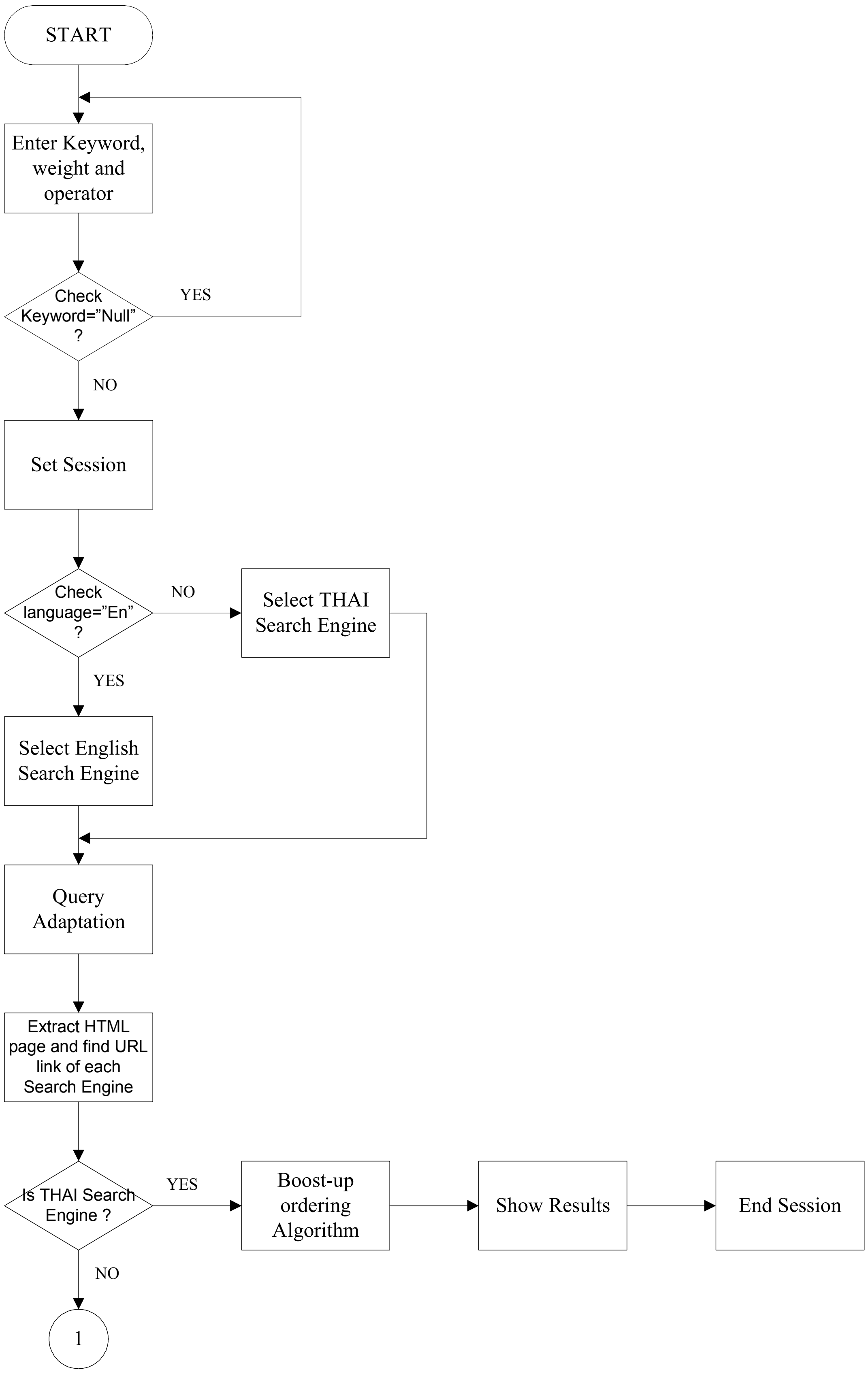

START
Enter Keyword, weight and operator
Check Keyword="Null" ?
YES
NO
Set Session
Check language="En" ?
NO
Select THAI Search Engine
YES
Select English Search Engine
Query Adaptation
Extract HTML page and find URL link of each Search Engine
Is THAI Search Engine ?
YES
Boost-up ordering Algorithm
Show Results
End Session
NO
1

Figure 4.2 Flowchart of the prototype.

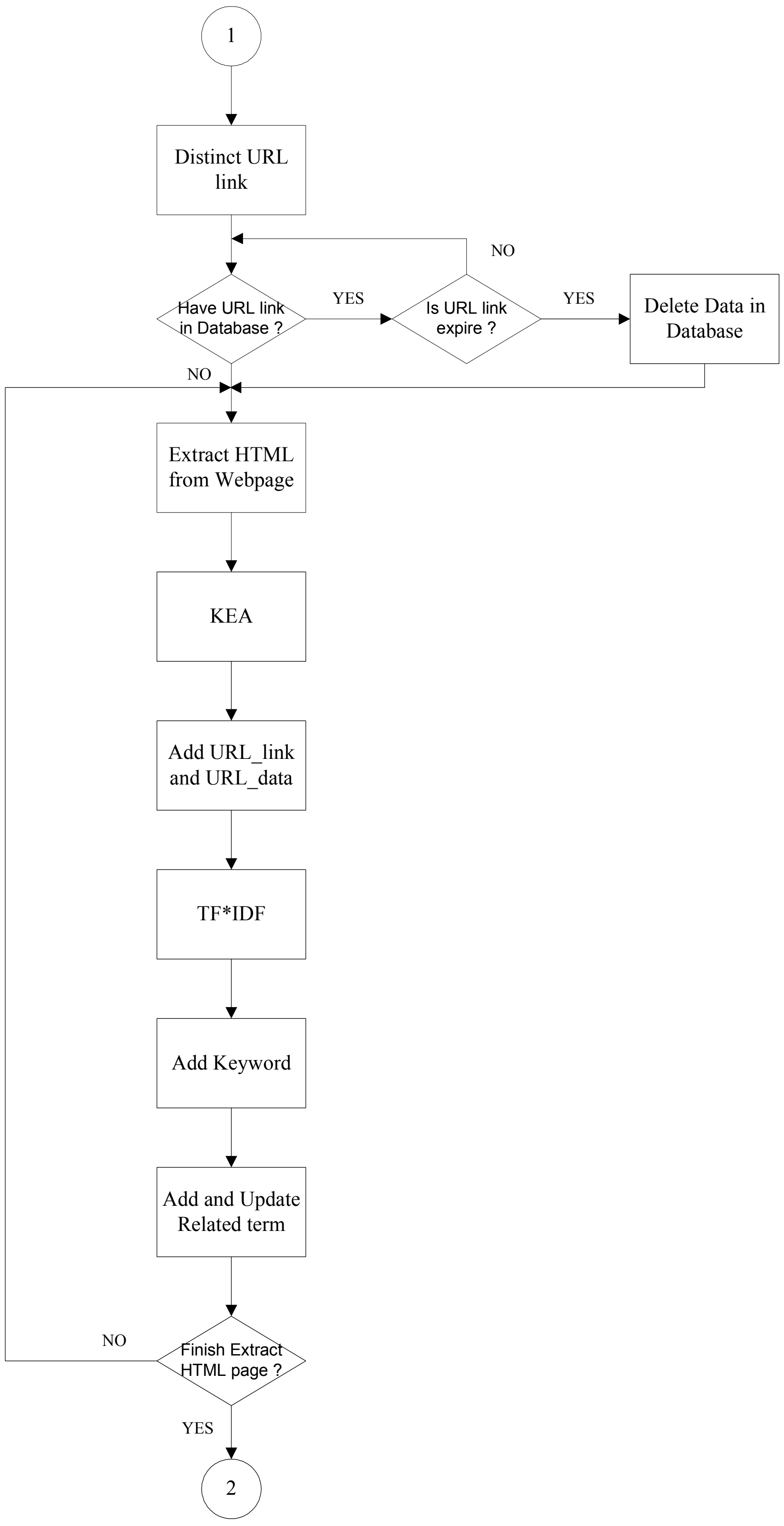

1
Distinct URL link
NO
Have URL link in Database ?
YES
Is URL link expire ?
YES
Delete Data in Database
NO
Extract HTML from Webpage
KEA
Add URL_link and URL_data
TF*IDF
Add Keyword
Add and Update Related term
NO
Finish Extract HTML page ?
YES
2

Figure 4.2 (Cont.)

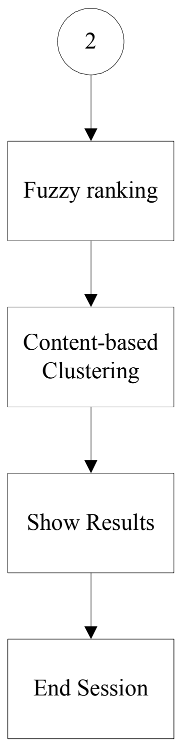


Figure 4.2 (Cont.)

.

The Figure 4.2 illustrates relevancy between processes and the flow of data around the system which are:

Enter keyword, weight and operator

This process is a user interface that provides communication between web users and the metasearch engine using fuzzy logic and content-based clustering system. It pass a user's query, weight of each query and Boolean operator to the metasearch engine.

Check keyword="Null"

This step is a checking process. If the keyword is null that means web users do not insert any terms into the system. The metasearch engine will return a message to users and wait until web users insert a keyword into the system. That message is “you do not insert the keywords, please insert keyword that you want to search”.

Set session

This module is prepared to handle with multi-users environment. The metasearch will generate the unique session ID for user query. So, web user can send the query to the system simultaneously.

Check language=”En”

The system employs keyphrase extraction to assist in fuzzy ranking and content-based clustering. Which use KEA to extract important keyword in homepage contents. However, KEA does not support THAI language. Therefore, this process will check the users‘ query that the keyword is THAI language query or English language query.

Select THAI search engine

The metasearch engine provides THAI term query. That means web users can search in THAI language. The system will pass the users' query to THAI search engines which are Sanook and Thaiseek.

Select English search engine

In english language query, the system passes the users‘ query to five effective search engines which are Google, Alltheweb, Altavista, Wisenut and Hotbot. These are top five from the size showdown [35] that compares nine search engines, with MSN search and HotBot representing the Inktomi database and used 25 small single word queries. This is also described in section 2.7.

Query adaptation

The query adaptation module is responsible for adapting the users' query to fit into each targeted search engine in the system. The query for each search engine is different by its server, method and variables. The server varies in host name, server port and remote path. The method is varied by GET and POST method which is PHP input method used by almost search engine. The variables are the parameters and their values that will be sent to each search engine.

Extract HTML page and find URL link of each search engine

This module receives the resultant web pages from all search engines. Since the web pages are cluttered documents that have too many unwanted data on the pages, it is necessary to parse the pages to bring out just only wanted data, with URL, title and descriptions. To do so, the system use the separated parsing submodule to parse the pages for each search engine because different search engines have different page formats.

Is THAI search engine

This step is a checking process that examines results from targeted search engine. If the results are returned from THAI search engine, this module will pass all results to the boost-up ordering algorithm for processing.

Boost-up ordering algorithm

This process reranks all of the results from THAI search engine by the boost-up ordering method, which is described in section 3.4. The reranked results are the input data for the content extraction module and the content-based clustering module.

The algorithm of the boost-up ordering method works as follows:

- It browses through the parsed results in the temporary database to find the duplicated URL.

- Then, it merges parsed results with duplicated URL and selects data of the better ranked ones. If the system is implemented by PHP.

Distinct URL link

This process will delete URL link that are duplicated. The system will select just one URL link that is duplicated and delete another URL link.

Have URL link in database

This module will examine URL link from targeted search engine, if the system find URL link in the database, it will not extract data from webpage. This process will decrease time in processing the system.

Is URL link expire

This step will check the date of URL link in the database. If URL link is over 3 month, it becomes dead and broken link. Therefore the system will extract data from webpage again and delete old data in the database.

Delete data in the database

If URL link expire module examines that URL link is over 3 month, all information of URL link in the database will be sent to delete data in the database module.

Extract HTML from webpage

The extract HTML from webpage module uses the parsed results which are the URL to request the HTML from each resultant site. This process takes longer time than other processes because it waits until all sites return the HTML to the system. This process makes the approach distinct from any others since the actual web pages are used for an analysis. However it also takes more time than other systems. The proposed approach is appropriate just for the search that does not need a real time searching but needs the post-retrieval analysing of the results.

KEA

The KEA module is the keyphrase extrator. It uses the HTML as the input data, and the output data are the content phrases of each HTML. The KEA is described in section 3.3. To adjust the extractor, the application program interface (API) is used to define the system values. This API is designed by KEA to allow the keyphrase extractor to be easily embedded in experimental or commercial software products.

Add URL_link and URL_data

After KEA module, the system will add all information of webpages in the database. The information will be added in the URL_link are URL link, Date and Time. The information will be added in the URL_data are URL link, Title and Description.

TF*IDF

In content-based clustering, weighted of term will be used for cosine similarity. Therefore, TF and IDF of each word must be calculated.

Add keyword

This module will add word, frequency and weight in the database. All of the data in keyword database will be used in fuzzy ranking and content-based clustering method.

Add and update related term

Fuzzy information retrieval is used to adopt a thesaurus which can be constructed by defining a related term or term-term correlation whose rows and columns are associated with the document collection. The basic idea is to expand the set of index terms in the query with related terms (obtained from thesaurus) such that additional relevant documents can be retrieved by the users' query. This module will add index1, index2 and value of term-term correlation.

Finish extract HTML page

This is a module for checking URL link from targeted search engine. The extract HTML from webpage module uses the parsed results which are the URL to request the HTML from each resultant site. After extract HTML page is complete, it will pass to fuzzy ranking module.

Fuzzy ranking

The fuzzy ranking module is an effective technique used to retrieve the information followed by terms that web users can weight with themselves. Not all terms are equally important in a query. Thus, it is important to allow users to indicate the relative importance of various terms by weighting them.

Content-based clustering

The content-based clustering engine performs two important functions. First, it uses the common-phrase heuristic clustering method to find the initial clusters. Second, it merges some initial clusters that some relationship by using the cluster merging method as described in the previous chapter. The final clusters are sent to the user interface component to be presented to the users.

Show result

This module shows the results from fuzzy ranking and content-based clustering. This is a final step that shows 15 links of results per page and cluster group which is related to the users' query. A novice user can perceive the advantage of the content-based clustering.

4.4 Data flow diagram

Dataflow diagram represents relevancy between processes and the flow of information around the system which are:

- Context diagram of metasearch engine using fuzzy logic and content-based clustering, Figure 4.3
- Dataflow diagram level 1 of metasearch engine using fuzzy logic and content-based clustering, Figure 4.4
- Dataflow diagram level 2 of search module, Figure 4.5
- Dataflow diagram level 2 of collect data module Figure, 4.6
- Dataflow diagram level 2 of fuzzy reranking and content-based clustering module, Figure 4.7
- Dataflow diagram level 3 of check keyword module Figure, 4.8
- Dataflow diagram level 3 of find URL link of each search engine module Figure, 4.9
- Dataflow diagram level 3 of check expire module Figure, 4.10
- Dataflow diagram level 3 of manipulate HTML module Figure, 4.11
- Dataflow diagram level 3 of add module Figure, 4.12
- Dataflow diagram level 3 of fuzzy ranking module Figure, 4.13
- Dataflow diagram level 3 of content-based clustering module Figure, 4.14

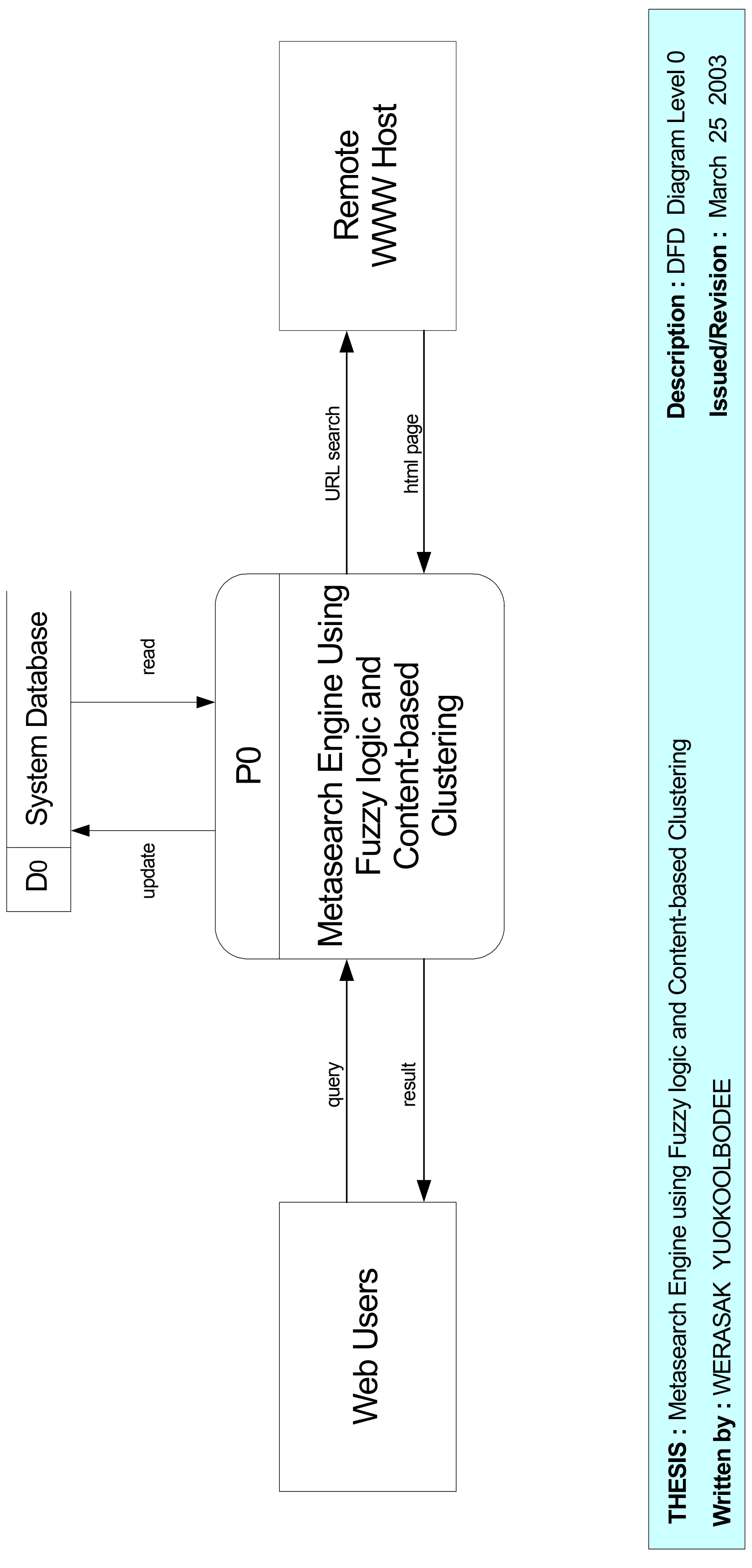

Web Users
query
result
P0
Metasearch Engine Using Fuzzy logic and Content-based Clustering
D0
System Database
read
update
URL search
html page
Remote WWW Host
THESIS : Metasearch Engine using Fuzzy logic and Content-based Clustering
Written by : WERASAK YUOKOOLBODEE
Description : DFD Diagram Level 0
Issued/Revision : March 25 2003

Figure 4.3 Context Diagram of Metasearch Engine using Fuzzy logic and Content-based Clustering

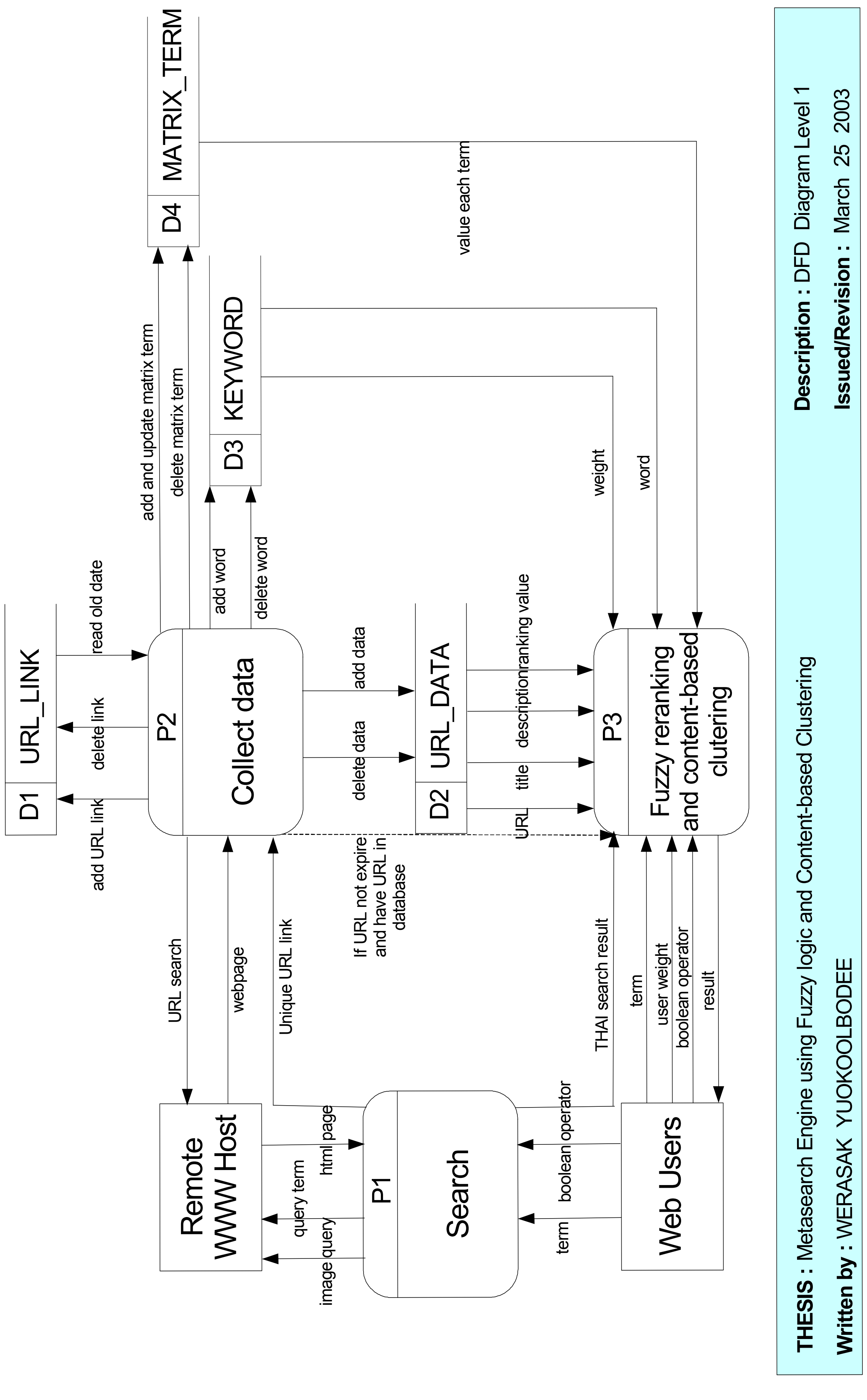


Figure 4.4 Data Flow Diagram level 1 of Metasearch Engine using Fuzzy logic and Content-based Clustering

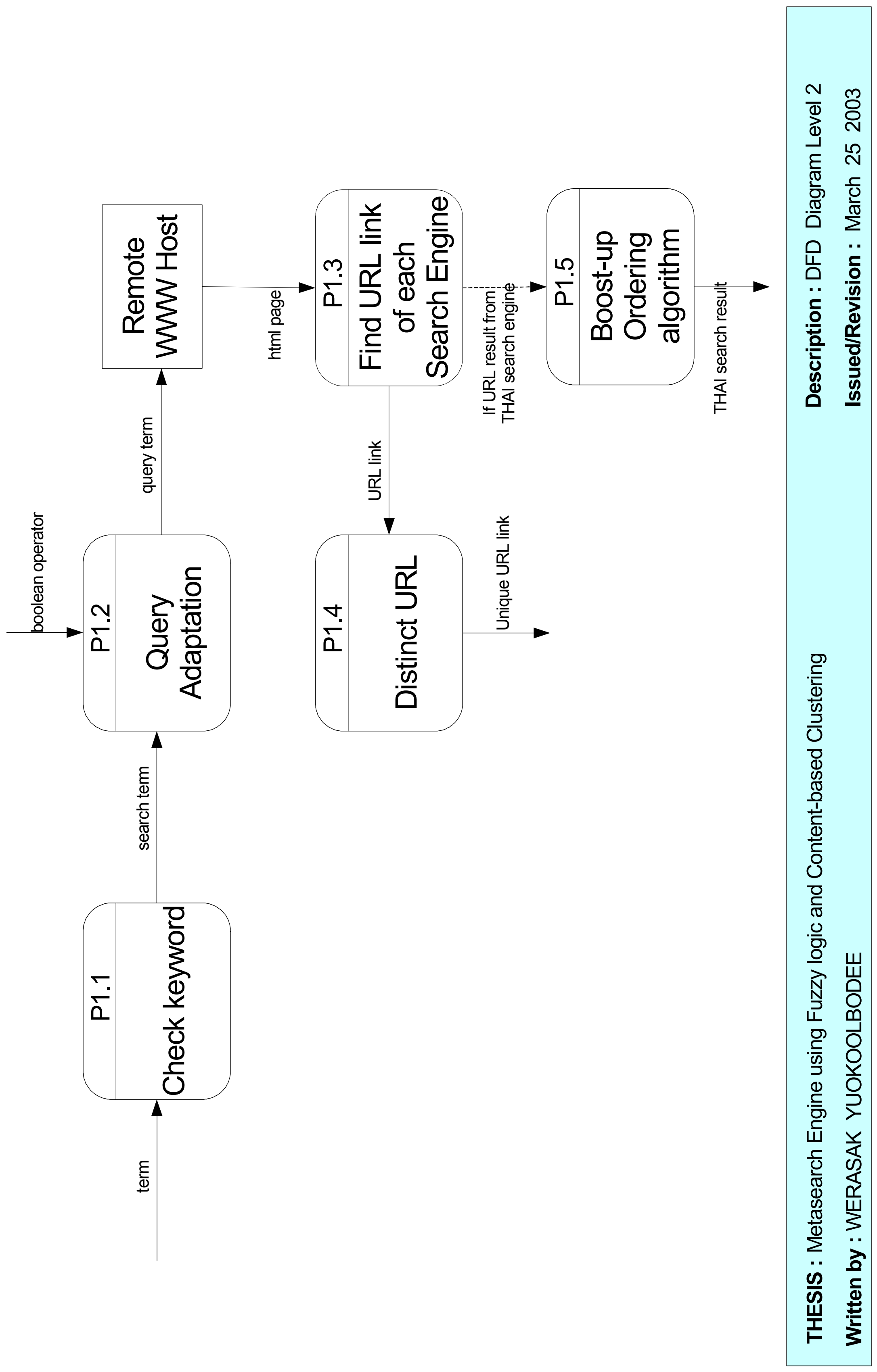

term
P1.1
Check keyword
search term
P1.2
Query Adaptation
boolean operator
query term
Remote WWW Host
html page
P1.3
Find URL link of each Search Engine
URL link
P1.4
Distinct URL
Unique URL link
If URL result from THAI search engine
P1.5
Boost-up Ordering algorithm
THAI search result
THESIS : Metasearch Engine using Fuzzy logic and Content-based Clustering
Written by : WERASAK YUOKOOLBODEE
Description : DFD Diagram Level 2
Issued/Revision : March 25 2003

Figure 4.5 Data Flow Diagram level 2 of Search Module

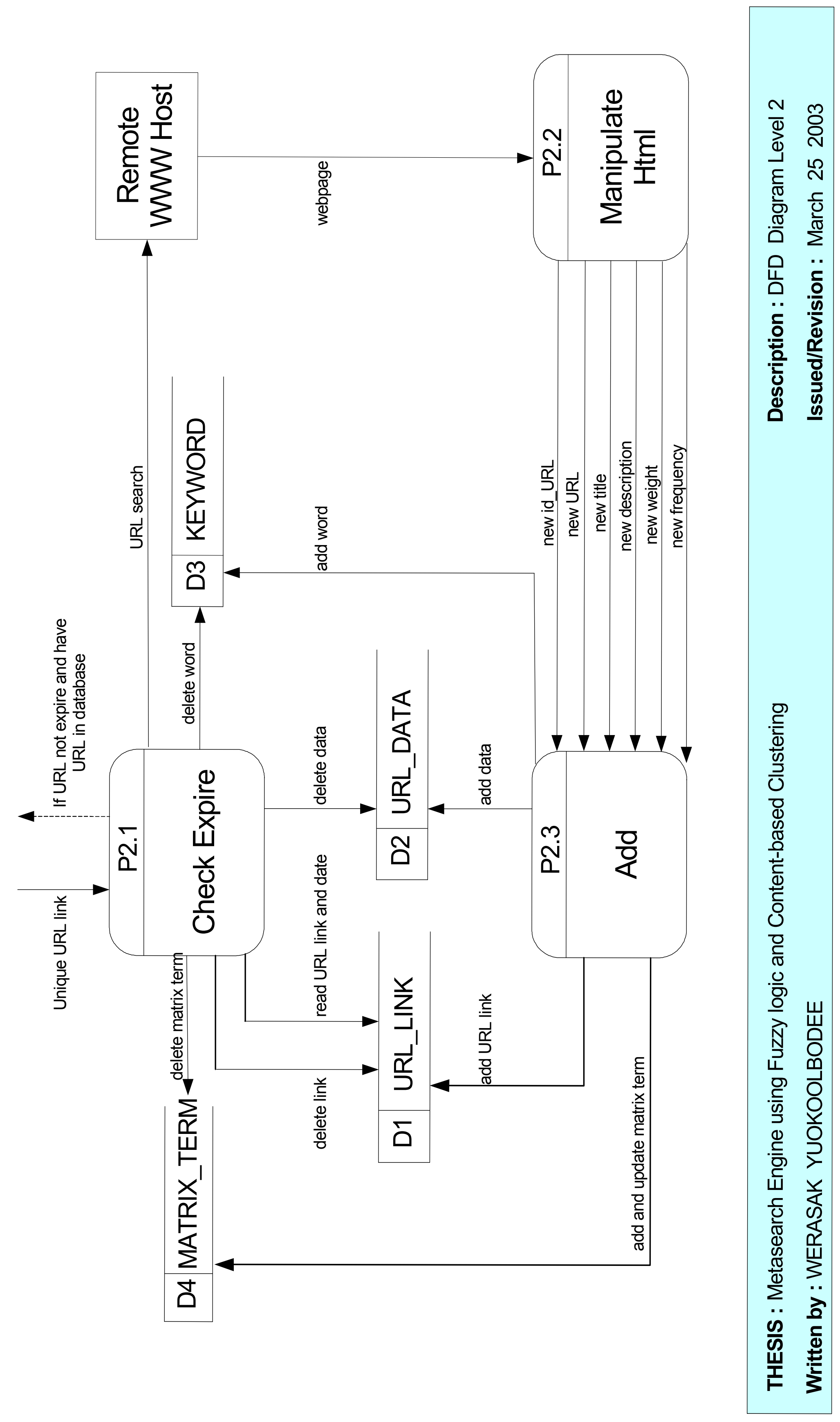
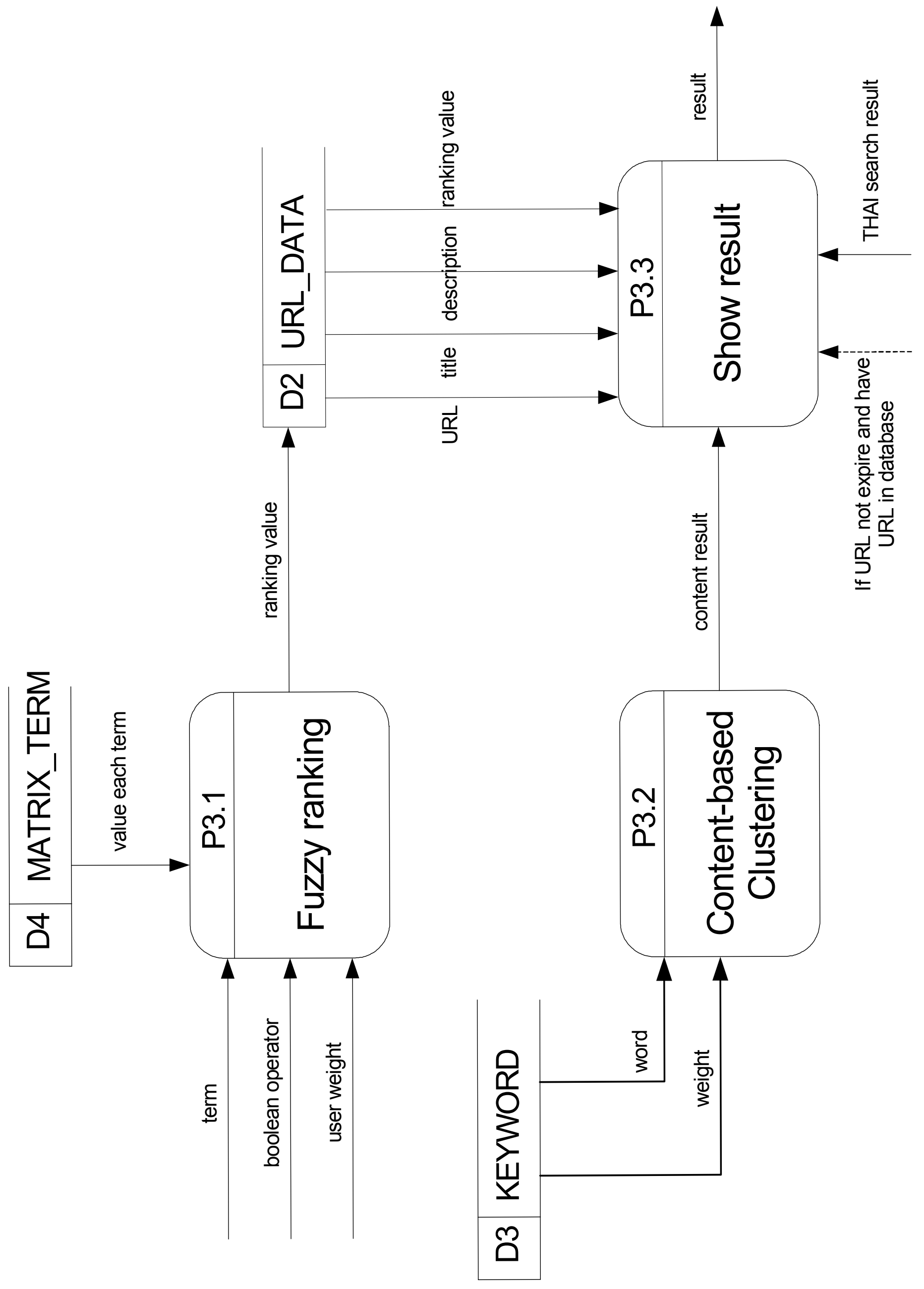


Figure 4.6 Data Flow Diagram level 2 of Collect data Module



THESIS : Metasearch Engine using Fuzzy logic and Content-based Clustering **Description :** DFD Diagram Level 2

Written by : WERASAK YUOKOOLBODEE **Issued/Revision :** March 25 2003

Figure 4.7 Data Flow Diagram level 2 of Fuzzy reranking and Content-based Clustering Module

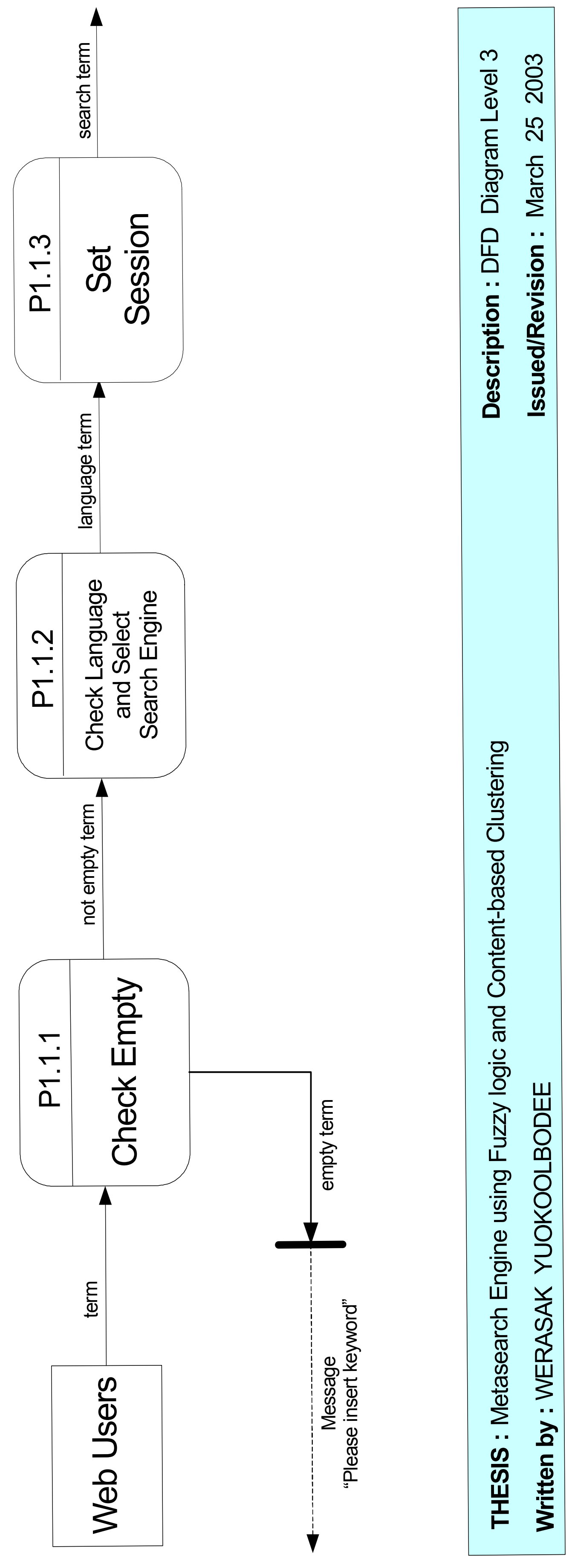
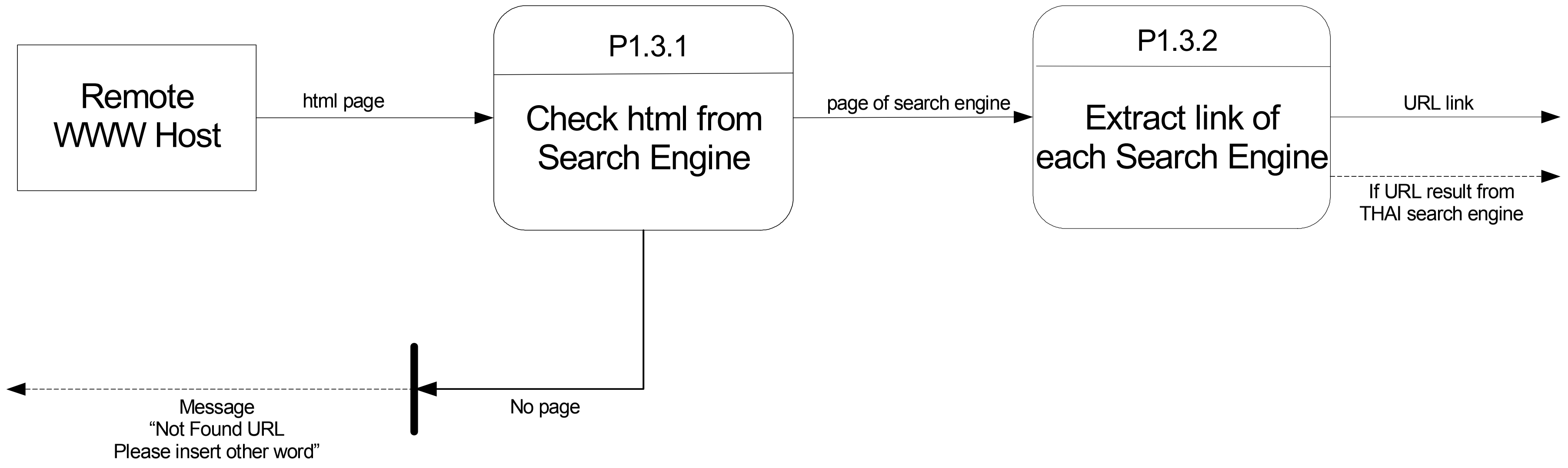

Web Users
term
P1.1.1
Check Empty
empty term
Message
“Please insert keyword”
not empty term
P1.1.2
Check Language and Select Search Engine
language term
P1.1.3
Set Session
search term
THESIS : Metasearch Engine using Fuzzy logic and Content-based Clustering
Written by : WERASAK YUOKOOLBODEE
Description : DFD Diagram Level 3
Issued/Revision : March 25 2003

Figure 4.7 Data Flow Diagram level 3 of Check Keyword Module


Remote WWW Host
html page
P1.3.1
Check html from Search Engine
page of search engine
P1.3.2
Extract link of each Search Engine
URL link
If URL result from THAI search engine
No page
Message "Not Found URL Please insert other word"

THESIS : Metasearch Engine using Fuzzy logic and Content-based Clustering **Description :** DFD Diagram Level 3

Written by : WERASAK YUOKOOLBODEE **Issued/Revision :** March 25 2003

Figure 4.8 Data Flow Diagram level 3 of Find URL link of each Search Engine Module

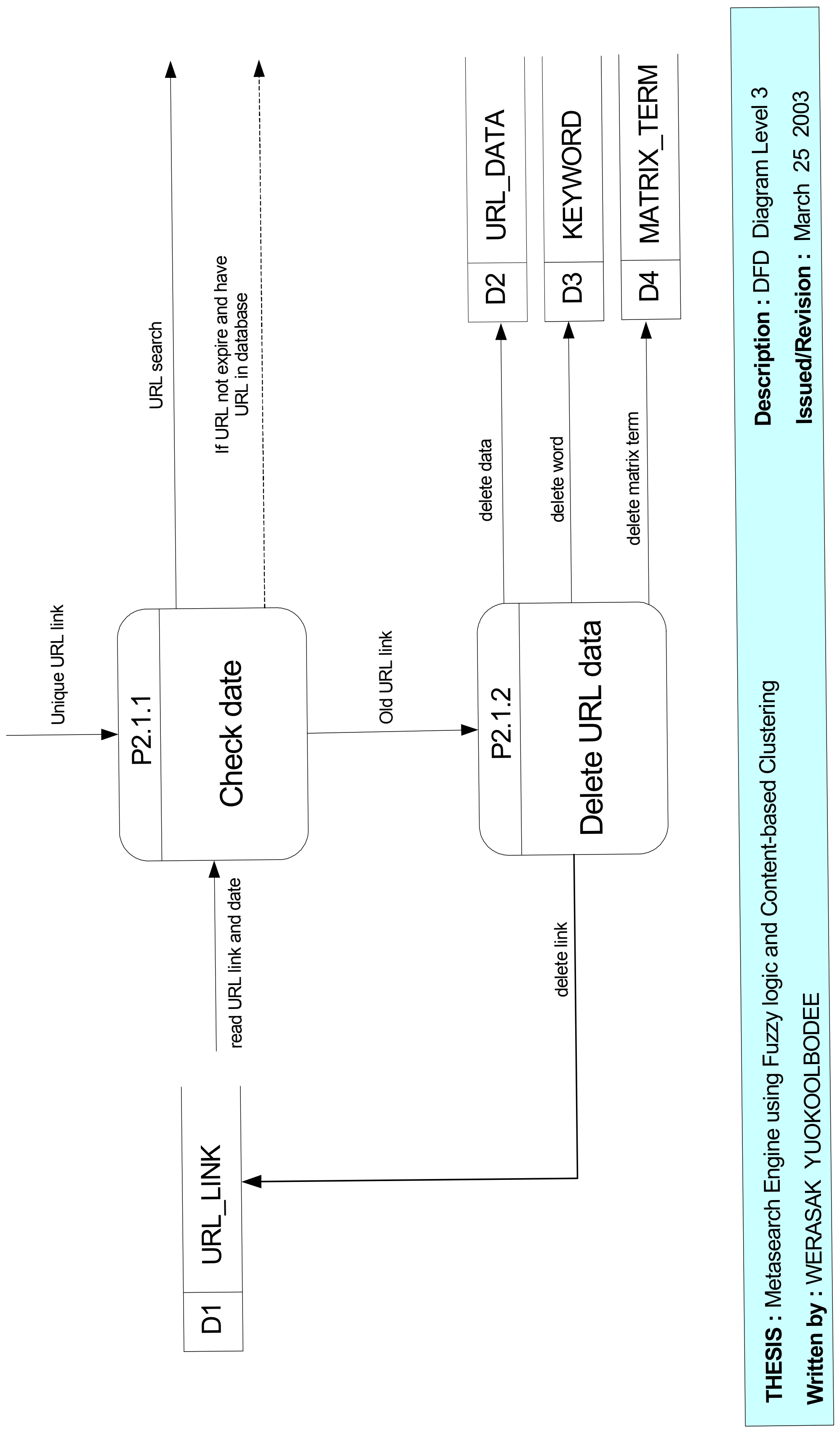
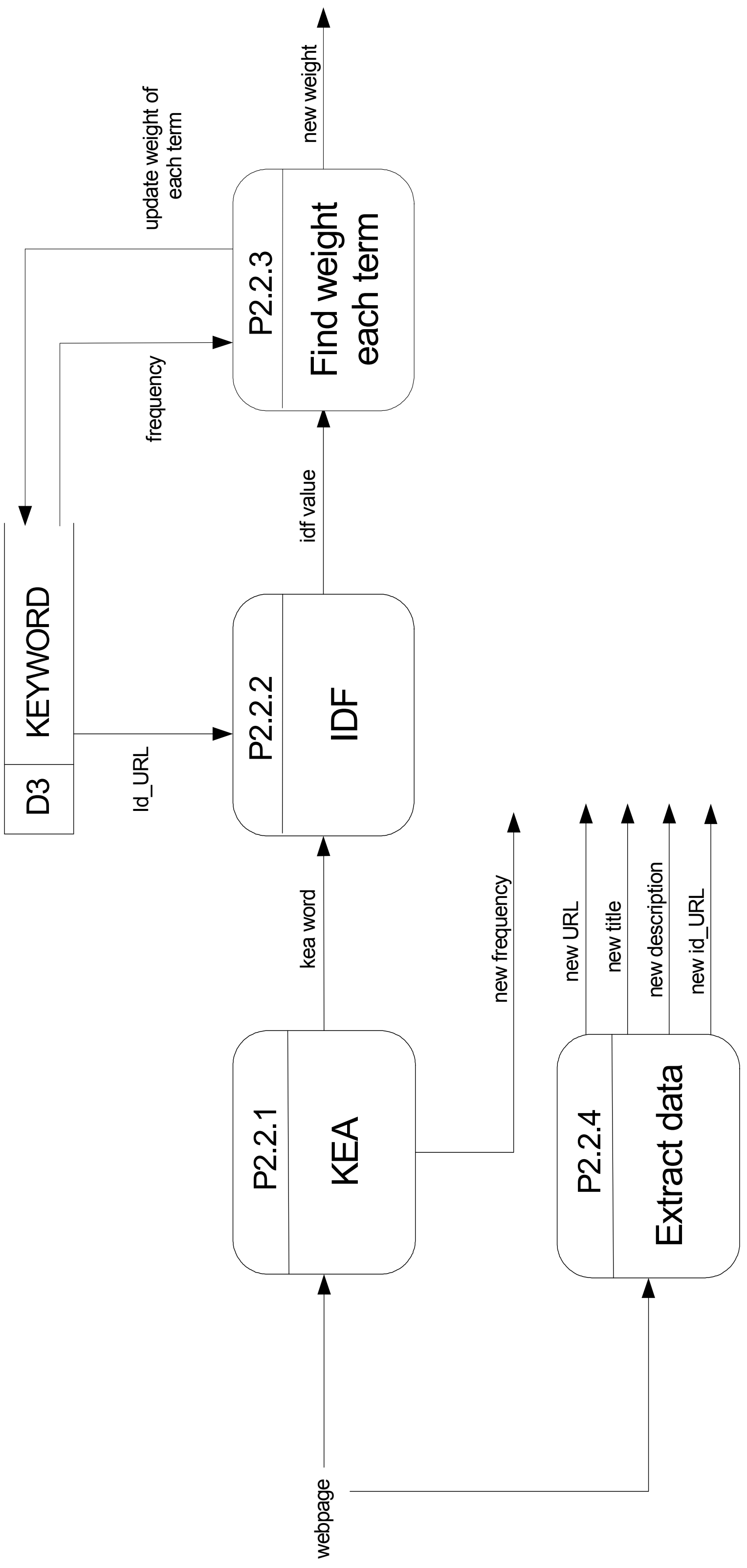

Unique URL link
P2.1.1
Check date
URL search
If URL not expire and have URL in database
read URL link and date
Old URL link
P2.1.2
Delete URL data
delete data
delete word
delete matrix term
delete link
D1 URL_LINK
D2 URL_DATA
D3 KEYWORD
D4 MATRIX_TERM
THESIS : Metasearch Engine using Fuzzy logic and Content-based Clustering
Written by : WERASAK YUOKOOLBODEE
Description : DFD Diagram Level 3
Issued/Revision : March 25 2003

Figure 4.10 Data Flow Diagram level 3 of Check Expire Module



THESIS : Metasearch Engine using Fuzzy logic and Content-based Clustering **Description :** DFD Diagram Level 3
Written by : WERASAK YUOKOOLBODEE **Issued/Revision :** March 25 2003

Figure 4.11 Data Flow Diagram level 3 of Manipulate Html Module

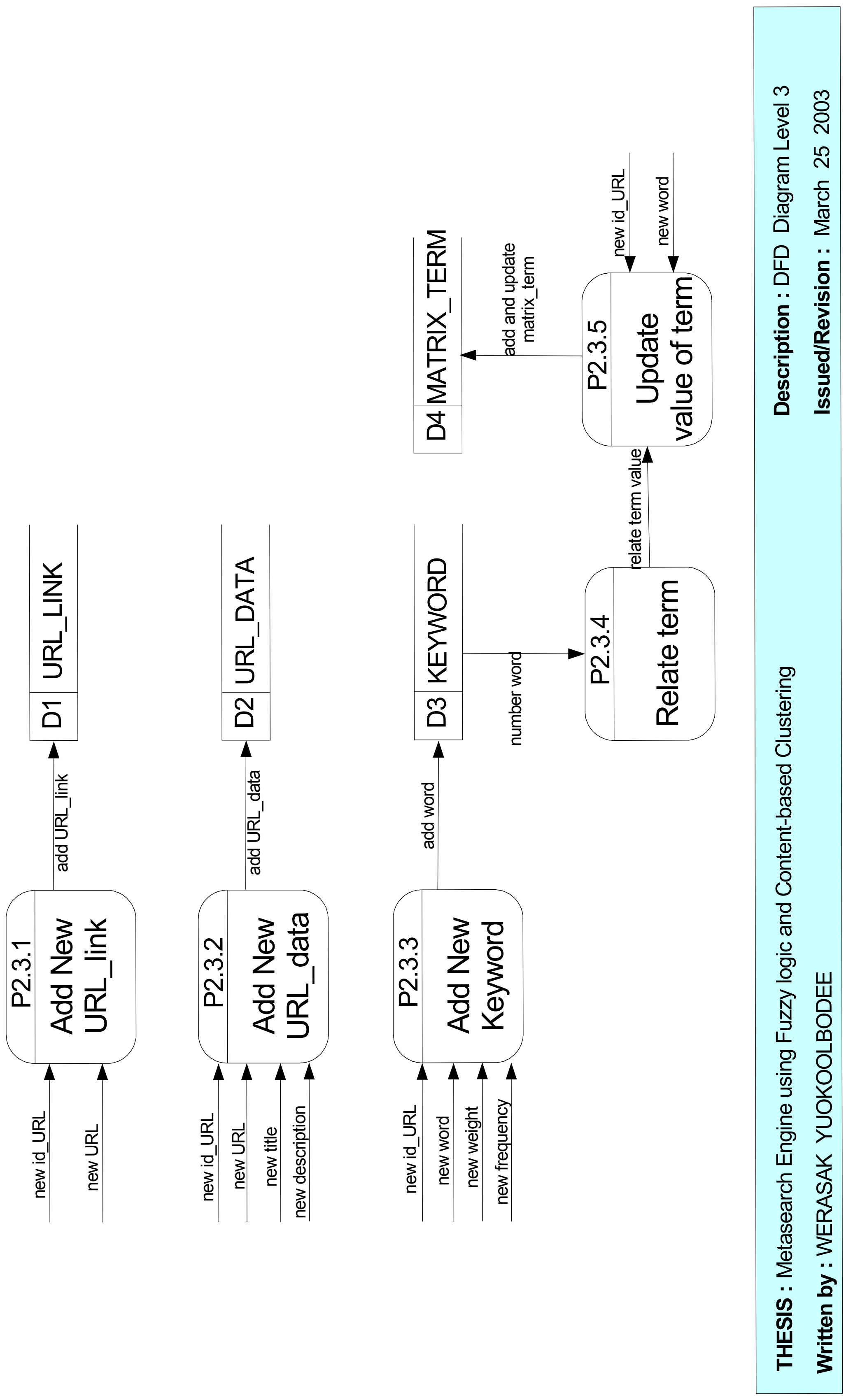
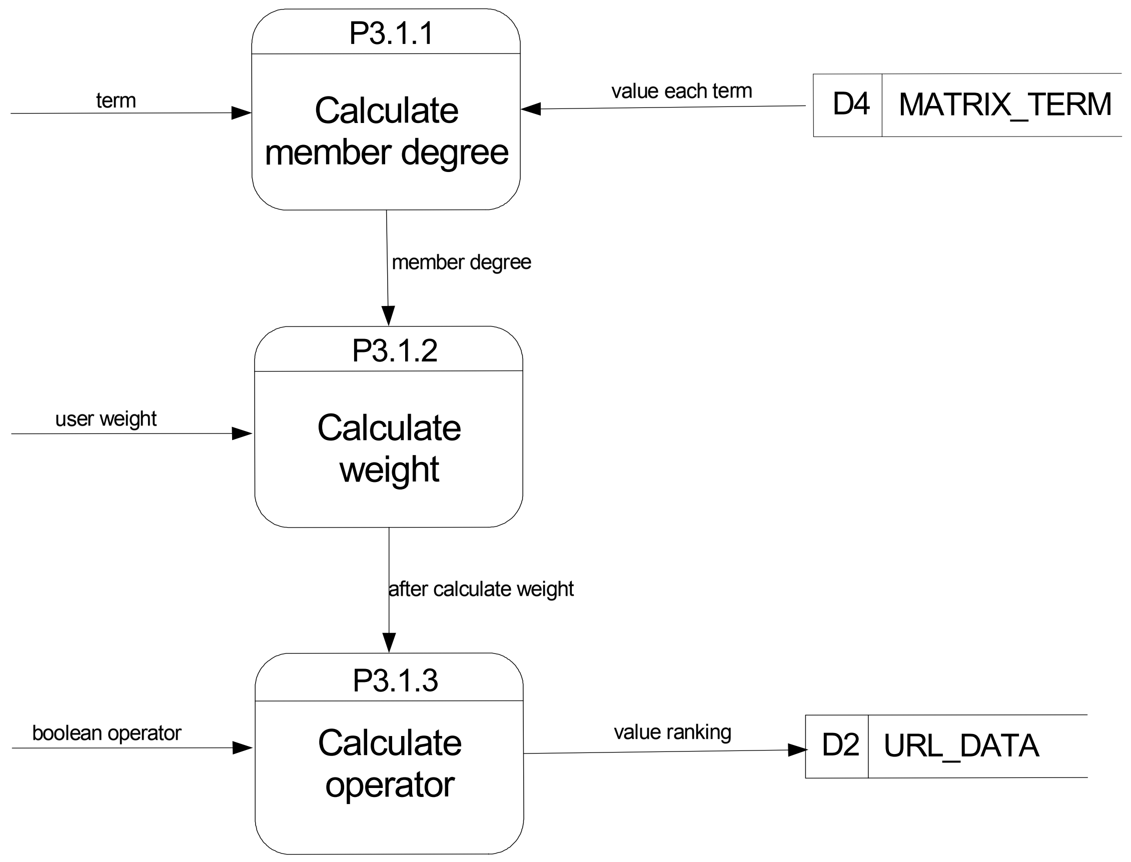


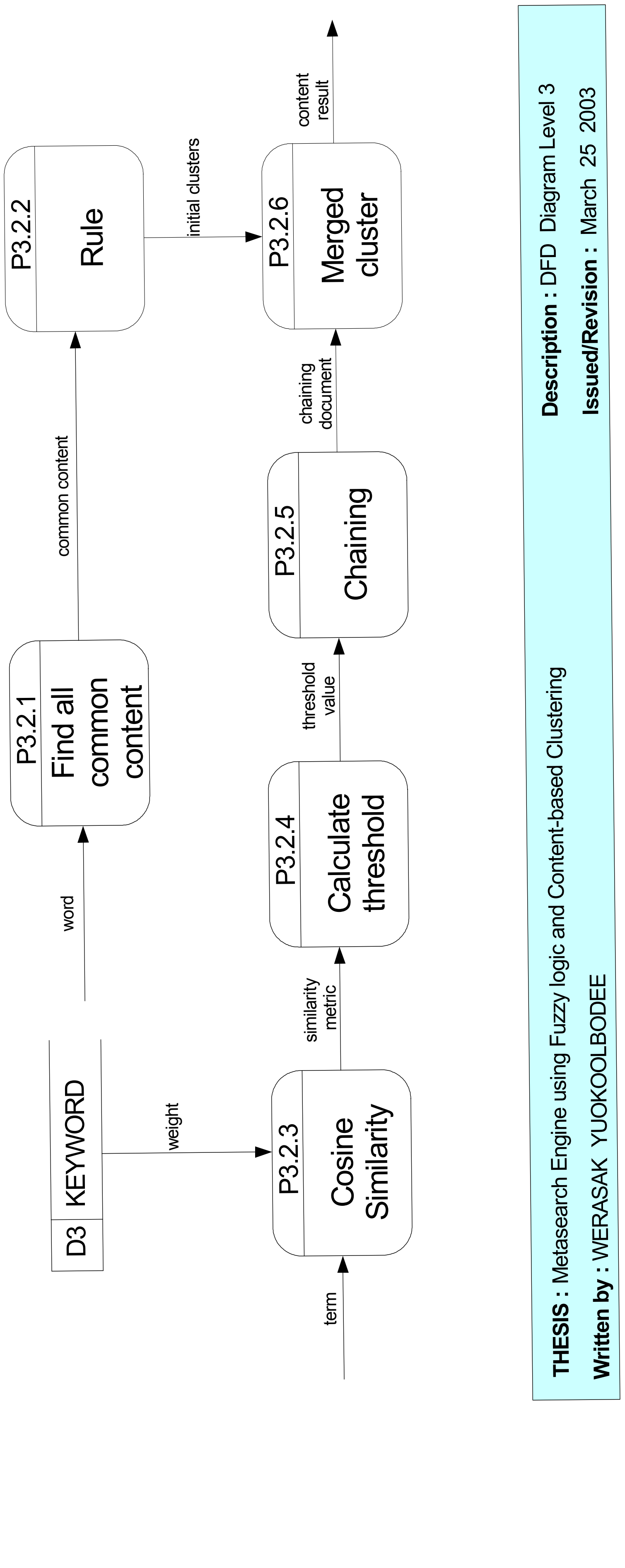
Figure 4.12 Data Flow Diagram level 3 of Add Module



THESIS : Metasearch Engine using Fuzzy logic and Content-based Clustering **Description :** DFD Diagram Level 3

Written by : WERASAK YUOKOOLBODEE **Issued/Revision :** March 25 2003

Figure 4.13 Data Flow Diagram level 3 of Fuzzy ranking Module



THESIS : Metasearch Engine using Fuzzy logic and Content-based Clustering **Description :** DFD Diagram Level 3

Written by : WERASAK YUOKOOLBODEE **Issued/Revision :** March 25 2003

Figure 4.14 Data Flow Diagram level 3 of Content-based Clustering Module

4.3 Data dictionary

Data dictionary represents all elements that relevant to the metasearch engine using fuzzy logic and content-based clustering. So both user and system analyst will understand in the system. Data dictionary are:

- Data process
- Data flow
- Data store

Data process

Table 4.3-4.7 show the details of data process, data flow and data store respectively.

Table 4.3 External Entities

Name	Web users
Description	The web users who want to retrieve information to satisfy their 's need which they can define weighting of keywords by themselves.
Inputs	• Result
Outputs	• Query

Name	Remote WWW Host
Description	The internet hosts that the system sent a query to and get data from these hosts.
Inputs	• URL search
Outputs	• Html page

Table 4.4 Data process

Process name	P0: Metasearch engine using fuzzy logic and content-based clustering
Description	For retrieve information to satisfy web users 's need
Inbound data flows	• Query • URL search
Outbound data flows	• Result • Html page
Module specification	Has three subsystem follow this : 1. Search Subsystem that that pass the query to the target search engines and extract URL link. 2. Collect data Subsystem that extract HTML from webpage and add new data to the database. 3. Fuzzy reranking and content-based clustering subsystem that reranking results from target search engine by using fuzzy logic and content-based clustering.

Process name	P1: Search
Description	Pass the query to the target search engines and extract URL link.
Inbound data flows	• Html page • Term • Boolean operator
Outbound data flows	• Query term • Unique URL link • THAI search result • Image query
Module specification	Has five function follow this : 1. Check keyword is checks the query from web users and examine the query that is in database or not. 2. Query adaptation is adapt a query from web users to appropriate target search engine. 3. Find URL link of search engine is extracts URL link of results that return from target search engine. 4. Distinct URL link is make URL link to be unique. 5. Boost-up ordering algorithm is use to reranking results from THAI search engine.

Process name	P2: Collect data
Description	Pass the query to the target search engines
Inbound data flows	• Web page • URL link • Read old date
Outbound data flows	• URL search • Add URL link • Delete link • Add and update matrix term • Delete matrix term • Add word • Delete word • Add data • Delete data • If URL not expire and have URL in database
Module specification	Has three function follow this : 1. Check expire is a module that check expire date of URL link in database. 2. Manipulate HTML is a module that extract important data from internet host.

	3. Add is a function that add all information to database.
Process name	P3: Fuzzy reranking and content-based clustering
Description	Reranking results from target search engine by using fuzzy logic and content-based clustering.
Inbound data flows	• Term • User weight • Boolean operator • Value each term • Word • Weight • URL • Title • Description • Ranking value • Thai search result • If URL not expire and have URL in database
Outbound data flows	• Result
Module specification	Has three function follow this : 1. Fuzzy ranking is use for reranking results from target search engine by fuzzy logic method. 2. Content-based clustering is a module that help a novice web users in searching information that satisfy their' s want. 3. Show result is a function that show the result from fuzzy ranking, boost-up ordering algorithm and content-based clustering method.

Process name	P1.1 Check keyword
Description	Check the query from web users and examine the query that is in database or not.
Inbound data flows	• Text query
Outbound data flows	• Search term
Module specification	Has subfunction follow this : 1. Check empty is a function that checks the query term from web users, if it empty, the system will show message and waiting until uses insert new query term. 2. Check language and select search engine is a function that checking the query term, it is THAI or English language to select appropriate search engine. 3. Set session is a function that starts session for multi-users query.

Process name	P1.2 Query adaptation
Description	Adapt the query from web users to appropriate target search engine.
Inbound data flows	• Search term • Boolean operator
Outbound data flows	• Query term
Module specification	-

Process name	P1.3 Find URL link of each search engine
Description	Extract URL link of results that return from target search engine.
Inbound data flows	• Html page
Outbound data flows	• URL link • If URL result from THAI search engine
Module specification	Has subfunction follow this : 1. Check HTML from search engine is a function that checking HTML page from target search engine, if search engine not return the results, the system will show message. 2. Extract link of each search engine is a function that extract URL link from results that return from target search engine.

Process name	P1.4 Distinct URL
Description	Make URL link to be unique.
Inbound data flows	• URL link
Outbound data flows	• Unique URL link

Module specification	If url1 = url2 then
	    Delete url2
	Else url1+url2
	end

Process name	P1.5 Boost-up ordering algorithm
Description	To reranking results from THAI search engine
Inbound data flows	• If URL result from THAI search engine
Outbound data flows	• Thai search result
Module specification	-

Process name	P2.1 Check expire
Description	To check expire date of URL link in database
Inbound data flows	• URL link • Read old date
Outbound data flows	• Delete link • Delete data • Delete word • Delete matrix term • URL search
Module specification	Has subfunction follow this : 1. Check date is a function that to examine expire date of URL link in database. 2. Delete URL date is a function that to delete all data in database, URL link has expired.

Process name	P2.2 Manipulate html
Description	Extract important data from internet host
Inbound data flows	• Web page
Outbound data flows	• New id_URL • New URL • New title • New description • New weight • New frequency
Module specification	Has subfunction follow this : 1. KEA is a function that is extract important word of webpage. 2. IDF is a function that calculated value of IDF for use in content-based clustering method. 3. Find weight each term is a function that calculated weight of each term. 4. Extract data is a function that is extract URL, title and description of each webpage.

Process name	P2.3 Add
Description	Add all information to database
Inbound data flows	• New id_URL • New URL • New title • New description • New weight • New frequency
Outbound data flows	• Add data • Add URL link • Add word • Add and update matrix term
Module specification	Has subfunction follow this : 1. Add new URL_link is a function that to add new URL link in database. 2. Add new URL_data is a function that to add new data in database. 3. Add new keyword is a function that to add new index term in database. 4. Relate term is a function that calculate term-term correlation. 5. Update value of term is a function that update new value of term in database.

Process name	P3.1 Fuzzy ranking
Description	To reranking results from target search engine by fuzzy logic method
Inbound data flows	• Value each term • Term • Boolean operator • User weight
Outbound data flows	• Ranking value
Module specification	Has subfunction follow this : 1. Calculate member degree is a function that to calculate member degree of each document. 2. Calculate weight is a function that to calculate weight of each document. 3. Calculate operator is a function that to calculate Boolean operator of each document.

Process name	P3.2 Content-based clustering
Description	To help a novice web users in searching information that satisfy their' s want.
Inbound data flows	• Word • Weight
Outbound data flows	• Content result
Module specification	Has subfunction follow this : 1. Find all common content is a function that to find all common content of all document. 2. Rule is a function that to checking rule of all document. 4. Cosine similarity is a function that to calculate cosine similarity of each document.

Process name	P3.3 Show result
Description	Show the result from fuzzy ranking, boost-up ordering algorithm and content-based clustering method.
Inbound data flows	• URL • Title • Description • Ranking value • Content result • THAI search result
Outbound data flows	• Result
Module specification	Has subfunction follow this : 1. Calculate threshold is a function that to calculate threshold. 2. Chaining is a function that to find chaining. 3. Merged cluster is a function that merging initial clusters and chaining document.

Process name	P1.1.1 Check empty
Description	To checking the query term from web users, if it empty, the system will show message and waiting until uses insert new query term.
Inbound data flows	• Term
Outbound data flows	• Not empty term
Module specification	If term="NULL" then show message "please insert keyword"

Process name	P1.1.2 Check language and select search engine
Description	To checking the query term, it is THAI or English language to select appropriate search engine.
Inbound data flows	• Not empty term
Outbound data flows	• Language term
Module specification	If term=”EN” then Select search engine Else Select THAI search engine End

Process name	P1.1.3 Set session
Description	Start session for multi-users query
Inbound data flows	• Language term
Outbound data flows	• Search term
Module specification	Start session

Process name	P1.3.1 Check html from search engine
Description	To checking HTML page from target search engine, if search engine not return the results, the system will show message.
Inbound data flows	• Html page
Outbound data flows	• Page of search engine • Message ”Not Found URL Please insert other word ”
Module specification	If html page=”NULL” then Show message “Not found Please insert other word” End

Process name	P1.3.2 Extract link of each search engine
Description	Extract URL link from results that return from target search engine
Inbound data flows	• Page of search engine
Outbound data flows	• URL link • If URL result from THAI search engine
Module specification	-

Process name	P2.1.1 Check date
Description	To examine expire date of URL link in database
Inbound data flows	• URL link Read old date
Outbound data flows	• URL search • if URL not expire and have URL in database old URL link

Module specification	If old_date>= 90 then
	    Call delete URL data module
	Else
	    Send to Remote WWW host
	end

Process name	P2.1.2 Delete URL data
Description	To delete all data in database, URL link has expired.
Inbound data flows	Old URL link
Outbound data flows	• Delete link • Delete data • Delete word • Delete matrix term
Module specification	

Process name	P2.2.1 KEA
Description	KEA is the Keyphrase Extraction to extract important word of webpage.
Inbound data flows	• Webpage
Outbound data flows	• Kea word • New frequency
Module specification	-

Process name	P2.2.2 IDF
Description	To calculated value of IDF for use in content-based clustering method.
Inbound data flows	Id_URL Kea word
Outbound data flows	IDF value
Module specification	-

Process name	P2.2.3 Find weight each term
Description	To calculated weight of each term
Inbound data flows	• IDF value • Frequency
Outbound data flows	• Update weight of each term • New weight
Module specification	-

Process name	P2.2.4 Extract data
Description	To extract URL, title and description of each webpage.
Inbound data flows	• Webpage
Outbound data flows	• New URL • New title • New description • New id_URL
Module specification	-

Process name	P2.3.1 Add New URL_link
Description	To add new URL link in database
Inbound data flows	• New id_URL • New URL
Outbound data flows	• Add URL_link
Module specification	-

Process name	P2.3.2 Add New URL_data
Description	To add new data in database
Inbound data flows	• New id_URL • New URL • New title • New description
Outbound data flows	• Add URL_data
Module specification	-

Process name	P2.3.3 Add New Keyword
Description	To add new index term in database
Inbound data flows	• New id_URL • New word • New weight • New frequency
Outbound data flows	• Add word
Module specification	-

Process name	P2.3.4 Relate term
Description	To calculate term-term correlation
Inbound data flows	• Number word
Outbound data flows	• Relate term value
Module specification	-

Process name	P2.3.5 Update value of term
Description	Update new value of term in database
Inbound data flows	• Relate term value • New id_URL • New word
Outbound data flows	• Add and update matrix term
Module specification	-

Process name	P3.1.1 Calculate member degree
Description	To calculate member degree of each document
Inbound data flows	• Term • Value each term
Outbound data flows	• Member degree
Module specification	-

Process name	P3.1.2 Calculate weight
Description	To calculate weight of each document
Inbound data flows	• Member degree • User weight
Outbound data flows	• After calculate weight
Module specification	-

Process name	P3.1.3 Calculate operator
Description	To calculate Boolean operator of each document
Inbound data flows	• Boolean operator • After calculate weight
Outbound data flows	• Value ranking
Module specification	-

Process name	P3.2.1 Find all common content
Description	To find all common content of all document
Inbound data flows	• Word
Outbound data flows	• Common content
Module specification	-

Process name	P3.2.2 Rule
Description	To checking rule of all document
Inbound data flows	• Common content
Outbound data flows	• Initial cluster
Module specification	-

Process name	P3.2.3 Cosine Similarity
Description	To calculate cosine similarity of each document
Inbound data flows	• Term • Weight
Outbound data flows	• Similarity metric
Module specification	-

Process name	P3.2.4 Calculate threshold
Description	To calculate threshold
Inbound data flows	• Similarity metric
Outbound data flows	• Threshold value
Module specification	-

Process name	P3.2.5 Chaining
Description	To find chaining
Inbound data flows	• Threshold value
Outbound data flows	• Chaining document
Module specification	-

Process name	P3.2.6 Merged Cluster
Description	Merging initial clusters and chaining document
Inbound data flows	• Chaining document • Initial clusters
Outbound data flows	• Content result
Module specification	-

Data flow

Table 4.5 Data flow

Dataflow name	Term
Description	Searching query from users.
Where used	• Metasearch Engine using fuzzy logic and content-based clustering • Search • Fuzzy reranking and content-based clustering • Check keyword • Fuzzy ranking • Check empty • Calculate member degree • Cosine similarity
How Used	Input

Dataflow name	User weight
Description	Weighting of term query from web users.
Where Used	• Metasearch Engine using fuzzy logic and content-based clustering • Fuzzy reranking and content-based clustering • Fuzzy ranking • Calculate weight
How Used	Input

Dataflow name	Boolean operator
Description	Boolean operator between query terms from users.
Where Used	• Metasearch Engine using fuzzy logic and content-based clustering • Search • Fuzzy reranking and content-based clustering • Query Adaptation • Fuzzy ranking • Calculate operator
How Used	Input

Dataflow name	Result
Description	Results from the system to web users.
Where Used	• Metasearch Engine using fuzzy logic and content-based clustering • Fuzzy reranking and content-based clustering • Show result
How Used	Output

Dataflow name	Query term
Description	Query term from the system to each search engine.
Where Used	• Metasearch Engine using fuzzy logic and content-based clustering • Search • Query adaptation
How Used	Output

Dataflow name	URL search
Description	Request data from the system to internet hosts.
Where Used	• Metasearch Engine using fuzzy logic and content-based clustering • Collect data • Check expire
How Used	Output

Dataflow name	Webpage
Description	Html data from internet to the system.
Where Used	• Metasearch Engine using fuzzy logic and content-based clustering • Collect data • Manipulate Html
How Used	Input

Dataflow name	Html page
Description	Html data of each search engine return to the system.
Where Used	• Metasearch Engine using fuzzy logic and content-based clustering • Search • Find URL link of each search engine
How Used	Input

Dataflow name	Unique URL link
Description	URL link
Where Used	• Search • Collect data • Distinct URL • Check expire • Check date
How Used	Input, Output

Dataflow name	THAI search result
Description	Thai search results from the system
Where Used	• Search • Fuzzy reranking and content-based clustering • Boost-up ordering algorithm • Show result
How Used	Input, Output

Dataflow name	Add URL link
Description	Add new URL link to database
Where Used	• Collect data • Add • Add new URL_link
How Used	Output

Dataflow name	Delete link
Description	Delete old URL link in database
Where Used	• Collect data • Check expire • Delete URL data
How Used	Output

Dataflow name	Read old date
Description	Read old date of URL link in database
Where Used	• Collect data • Check expire • Check date
How Used	Input

Dataflow name	Add and update matrix term
Description	Add and update new matrix term in database
Where Used	• Collect data • Add • Update value of term
How Used	Output

Dataflow name	Delete matrix term
Description	Delete old matrix term in database
Where Used	• Collect data • Check expire •
How Used	Output

Dataflow name	Add word
Description	Add new keyword in database.
Where Used	• Collect data • Add • Add new keyword
How Used	Output

Dataflow name	Delete word
Description	Delete expire URL link data in database.
Where Used	• Collect data • Check expire • Delete URL data
How Used	Output

Dataflow name	Add data
Description	Add new URL data in database.
Where Used	• Collect data • Add • Add new URL_data
How Used	Output

Dataflow name	Delete data
Description	Delete expire URL data in database
Where Used	• Collect data • Check expire • Delete URL data
How Used	Output

Dataflow name	If URL not expire and have URL in database
Description	when URL link not expire and have URL data in database will show result.
Where Used	• Collect data • Fuzzy reranking and content-based clustering • Check expire • Show result • Check date
How Used	Control flow

Name	URL
Description	Pull URL link to show result process
Where Used	• Fuzzy reranking and content-based clustering • Show result
How Used	Input

Dataflow name	Title
Description	Pull title of URL link to show result process
Where Used	• Fuzzy reranking and content-based clustering • Show result
How Used	Input

Dataflow name	Description
Description	Pull description of URL to show result process
Where Used	• Fuzzy reranking and content-based clustering • Show result
How Used	Input

Data name	Ranking value
Description	Pull ranking value of each document to show result
Where Used	• Fuzzy reranking and content-based clustering • Show result
How Used	Input

Dataflow name	Weight
Description	Extract weight of each term in database to calculate
Where Used	• Fuzzy reranking and content-based clustering • Content-based clustering • Cosine similarity
How Used	Input

Dataflow name	Word
Description	Extract word of each term in database to calculate.
Where Used	• Fuzzy reranking and content-based clustering • Content-based clustering • Find all common content
How Used	Input

Dataflow name	Value each term
Description	Pull value of term-term correlation in database
Where used	• Fuzzy reranking and content-based clustering • Fuzzy ranking • Value each term
How used	Input

Dataflow name	Search term
Description	Send term query to adaptation process
Where Used	• Check keyword • Query adaptation
How Used	Input, Output

Dataflow name	URL link
Description	Send URL link of webpage to find unique URL link.
Where Used	• Find URL link of each Search Engine • Distinct URL
How Used	Input, Output

Dataflow name	If URL result from THAI search engine
Description	If URL result from THAI search engine to reranking.
Where Used	• Find URL link of each search engine • Boost-up ordering algorithm
How Used	Input, Output

Dataflow name	New id_URL
Description	Send new id_URL for add to database
Where Used	• Manipulate Html • Add • Add new URL_link • Add new URL_data • Add new URL_keyword • Update value of term
How Used	Input, Output

Dataflow name	New URL
Description	Send new URL link for add to database
Where Used	• Manipulate Html • Add • Add new URL_link • Add new URL_data
How Used	Input, Output

Dataflow name	New title
Description	New title
Where Used	• Manipulate Html • Add • Add new URL_data
How Used	Input, Output

Dataflow name	New description
Description	Send new description of URL link add to database
Where Used	• Manipulate Html • Add • Add new URL_data •
How Used	Input, Output

Dataflow name	New weight
Description	Send new weight of each term to calculate.
Where Used	• Manipulate Html • Add • Add new keyword
How Used	Input, Output

Dataflow name	New word
Description	Send new word for add to database
Where Used	• Manipulate Html • Add • Add new keyword • Update value of term
How Used	Input, Output

Dataflow name	New frequency
Aliases	-
Where Used	• Manipulate Html • Add • Add new keyword
How Used	Input, Output

Dataflow name	Content result
Description	Send content results to show result process
Where Used	• Content-based clustering • Show result
How Used	Input, Output

Dataflow name	Not empty term
Description	If query not empty will send to next process
Where Used	• Check empty • Check language and select search engine
How Used	Input, Output

Dataflow name	Language term
Description	Send query to start session process
Where Used	• Set session • Check language and select search engine
How Used	Input, Output

Dataflow name	Message “Please insert keyword”
Description	Send message to web users.
Where Used	• Check empty
How Used	Control flow

Dataflow name	Page of search engine
Description	Send Html page of each search engine to next process.
Where Used	• Check html from search engine • Extract link of each search engine
How Used	Input, Output

Dataflow name	Message ”Not Found URL please insert other word”
Description	Send message to web users.
Where Used	• Check html from search engine
How Used	Control flow

Dataflow name	Old URL link
Description	Send old URL link id to next process
Where Used	• Check date • Delete URL data
How Used	Input, Output

Dataflow name	Kea word
Description	Send important word to calculate
Where Used	• KEA • IDF
How Used	Input, Output

Dataflow name	Idf value
Description	Send idf value to calculate weight of each term.
Where Used	• IDF • Find weight of each term
How Used	Input, Output

Dataflow name	Update weight of each term
Description	Update new weight of each term to database
Where Used	• Find weight each term
How Used	Output

Dataflow name	Frequency
Description	Pull frequency of each term to calculate
Where Used	• Find weight each term
How Used	Input

Dataflow name	Id_URL
Description	Pull id_URL of each term to calculate IDF
Where Used	• IDF
How Used	Input

Dataflow name	Number word
Description	Pull numbers of all word to calculate.
Where Used	• Relate term
How Used	Input

Dataflow name	Relate term value
Description	Send relate term to calculate
Where Used	• Relate term • Update value of term
How Used	Input, Output

Dataflow name	Member degree
Description	Send member degree to calculate weight
Where Used	• Calculate member degree • Calculate weight
How Used	Input, Output

Name	After calculate weight
Description	Send weight value to calculate operator
Where Used	• Calculate weight • Calculate operator
How Used	Input, Output

Dataflow name	Common content
Description	Send common content to calculate initial clusters
Where Used	• Find all common content • Rule
How Used	Input, Output

Dataflow name	Initial clusters
Description	Send initial cluster to merged cluster process
Where Used	• Rule • Merged cluster
How Used	Input, Output

Dataflow name	Similarity metric
Description	Send similarity metric to find threshold value
Where Used	• Calculate threshold • Cosine similarity
How Used	Input, Output

Dataflow name	Threshold value
Aliases	-
Where Used	• Calculate threshold • Chaining
How Used	Input, Output

Dataflow name	Chaining document
Description	Send chaining document to find content result.
Where Used	• Chaining • Merged cluster
How Used	Input, Output

Data store

Table 4.6 Data store

Data store name	D1 URL_link
Description	To keep webpage
Inbound data flows	• Add URL link • Delete link
Outbound data flows	• Read old date

Data store name	D2 URL_data
Description	To keep information in webpage
Inbound data flows	• Delete data • Add data
Outbound data flows	• URL • Title • Description • Ranking value

Data store name	D3 Keyword
Description	To keep content about webpage
Inbound data flows	• Add word • Delete word
Outbound data flows	• Weight • Word

Data store name	D4 Matrix_term
Description	To keep all correlate term
Inbound data flows	• Add and update matrix term • Delete matrix term
Outbound data flows	• Value each term

4.6 File structure

Table 4.7 File structure of URL_link

Attribute		Type	Size	Constrain	Description
PK	ID_URL	Integer	9	Not Null	URL code
	URL	Varchar	200	Not Null	URL Name
	DATE	Date	8	Not Null	Date of each URL
	TIME	Time	4	Not Null	Time of each URL

Table 4.8 File structure of URL_data

Attribute		Type	Size	Constrain	Description
PK,FK	ID_URL	Integer	9	Not Null	URL Code
	URL	Varchar	200	Not Null	URL Name
	TITLE	Date	100	Not Null	Title of each URL
	DESCRIPTION	Time	400	Not Null	Description of each URL
	RANKING	Float	8	Null	Ranking of each webpage

Table 4.9 File structure of keyword

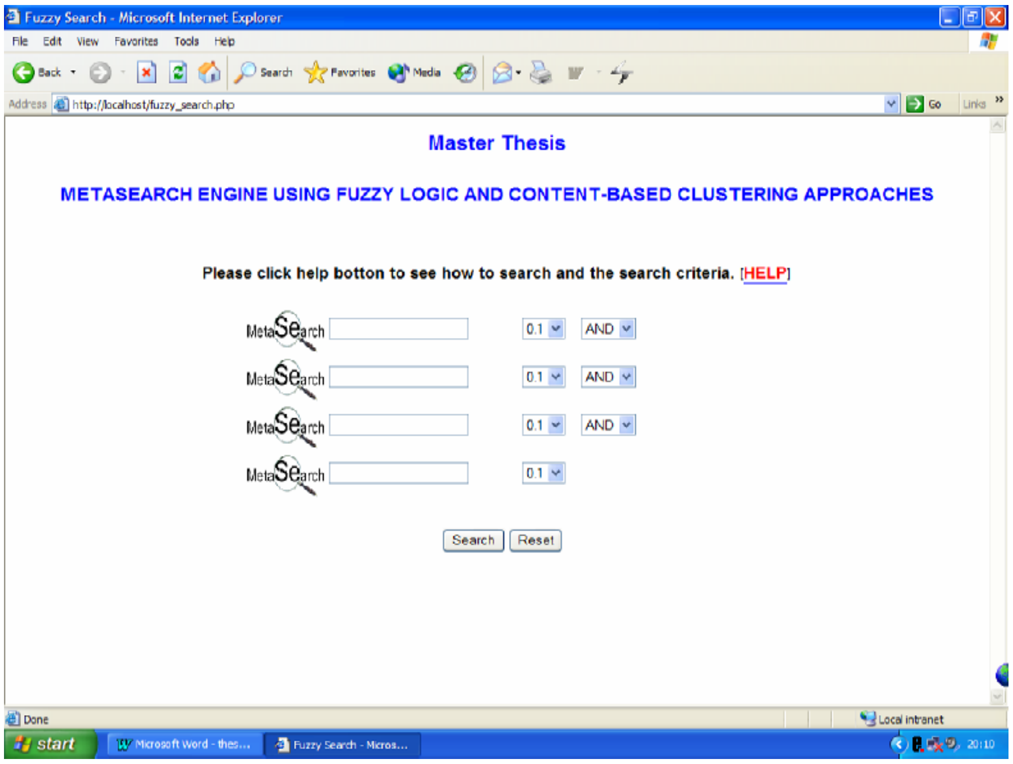
Attribute		Type	Size	Constrain	Description
PK,FK	ID_URL	Integer	9	Not Null	URL code
PK	WORD	Varchar	50	Not Null	Index of each webpage
	FREQUENCY	Integer	8	Not Null	Frequency of each index
	WEIGHT	Float	8	Not Null	Weight of each index

Table 4.10 File structure of matrix term

Attribute		Type	Size	Constrain	Description
PK,FK	ID_URL	Integer	9	Not Null	URL code
PK	Index1	Varchar	50	Not Null	Index1 of each webpage
PK	Index2	Varchar	50	Not Null	Index2 of each webpage
	Value	Float	8	Not Null	Value of term-term correlation

4.7 Implementation of user interface

The user interface is a component that provides communication between a user and the metasearch engine using fuzzy logic and content-based clustering. It pass a web users' s query to the metasearch engine. Also, it receives the results from fuzzy ranking method and the clustered results from the content-based clustering method to show web users. Figure 4.15 illustrates the user interface.


Master Thesis

METASEARCH ENGINE USING FUZZY LOGIC AND CONTENT-BASED CLUSTERING APPROACHES

Please click help botton to see how to search and the search criteria. [HELP]

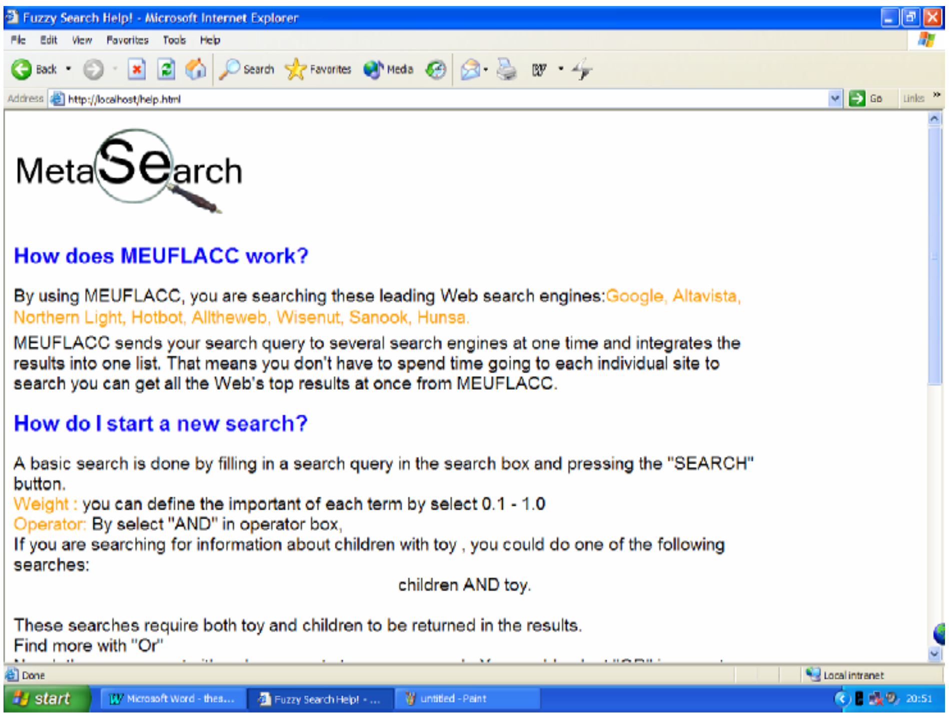
Fuzzy Search - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://localhost/fuzzy_search.php
Go Links
MetaSearch
0.1
AND
Search
Reset
Done
Local intranet
start
Microsoft Word - thes...
Fuzzy Search - Micros...
20:10

Figure 4.15 The users interface of MEUFLACC

The searching interface allows the web users to put their queries to the system by:

- Putting the queries in searching from that the system can receive 1- 4 query terms.
- Defining weight of each term that the values are 0.1-1.0.
- Selecting Boolean operator, AND, OR.
- Putting the search button to starting the searching process.

When the novice users do not know what should they do, the help button will assist them by showing a page that illustrates what is the system, how to search and the search criteria. The help page is shown below in Figure 4.16.


Fuzzy Search Help! - Microsoft Internet Explorer

http://localhost/help.html

MetaSearch

How does MEUFLACC work?

By using MEUFLACC, you are searching these leading Web search engines:Google, Altavista, Northern Light, Hotbot, Alltheweb, Wisenut, Sanook, Hunsa.

MEUFLACC sends your search query to several search engines at one time and integrates the results into one list. That means you don't have to spend time going to each individual site to search you can get all the Web's top results at once from MEUFLACC.

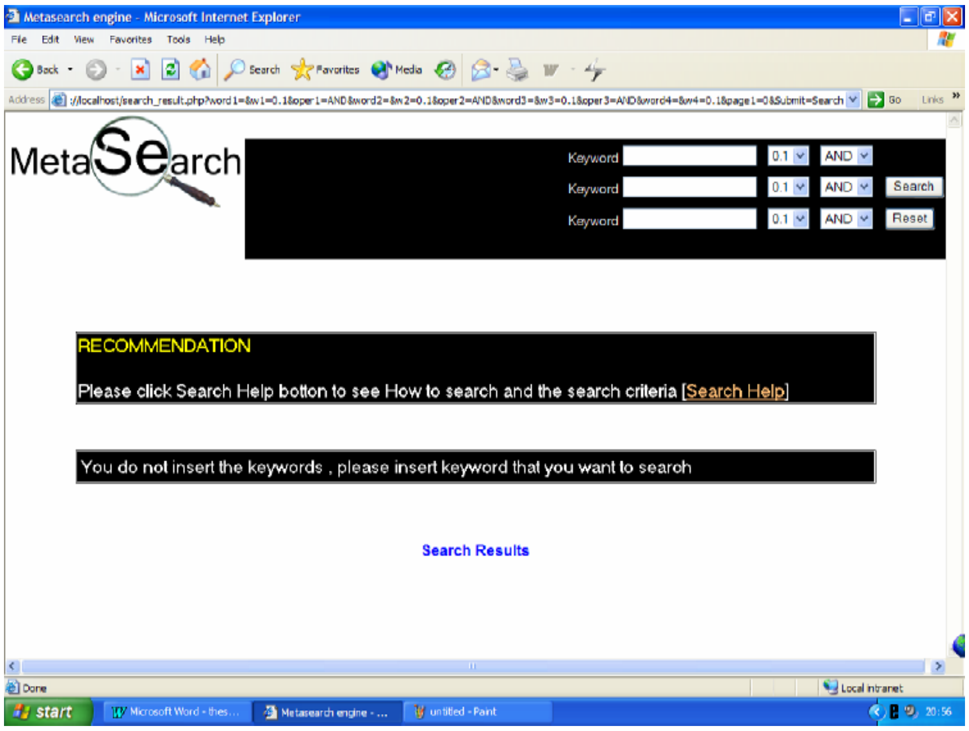
How do I start a new search?

A basic search is done by filling in a search query in the search box and pressing the "SEARCH" button.
Weight : you can define the important of each term by select 0.1 - 1.0
Operator: By select "AND" in operator box,
If you are searching for information about children with toy , you could do one of the following searches:

children AND toy.

These searches require both toy and children to be returned in the results.
Find more with "Or"

Figure 4.16 The help page


Metasearch engine - Microsoft Internet Explorer

://localhost/search_result.php?word1=&w1=0.1&oper1=AND&word2=&w2=0.1&oper2=AND&word3=&w3=0.1&oper3=AND&word4=&w4=0.1&page1=0&Submit=Search

MetaSearch

Keyword		0.1	AND	
Keyword		0.1	AND	Search
Keyword		0.1	AND	Reset

RECOMMENDATION

Please click Search Help botton to see How to search and the search criteria [Search Help]

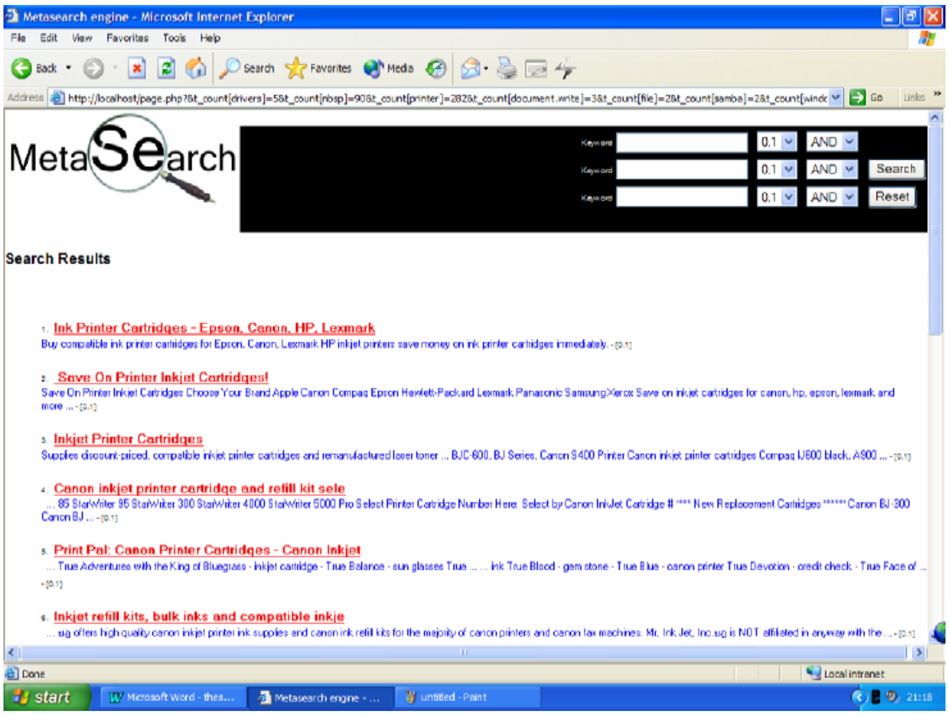
You do not insert the keywords , please insert keyword that you want to search

Search Results

Figure 4.17 The warning page

The searching interface will show the warning page when web users not put any query in the search form and click the search button. The warning page is shown in Figure 4.17.

The results will be separated into parts: first, the result is a ranked list page that is illustrated top list reranking by fuzzy logic method. Second, the result is the clustered resulting from content-based clustering method. Figure 4.18 shows the ranked list page and Figure 4.19 shown the clustered page.


Metasearch engine - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://localhost/page.php?&t_count[drivers]=5&t_count[nbsp]=90&t_count[printer]=282&t_count[document.write]=3&t_count[file]=2&t_count[samba]=2&t_count[windc
Go Links
MetaSearch
Keyword 0.1 AND
Keyword 0.1 AND Search
Keyword 0.1 AND Reset
Done
Local intranet
start
Microsoft Word - thes...
Metasearch engine - ...
untitled - Paint
21:18

Search Results

1. Ink Printer Cartridges - Epson, Canon, HP, Lexmark
Buy compatible ink printer cartridges for Epson, Canon, Lexmark HP inkjet printers save money on ink printer cartridges immediately. - [0.1]

2. Save On Printer Inkjet Cartridges!
Save On Printer Inkjet Cartridges Choose Your Brand Apple Canon Compaq Epson Hewlett-Packard Lexmark Panasonic Samsung Xerox Save on inkjet cartridges for canon, hp, epson, lexmark and more ... - [0.1]

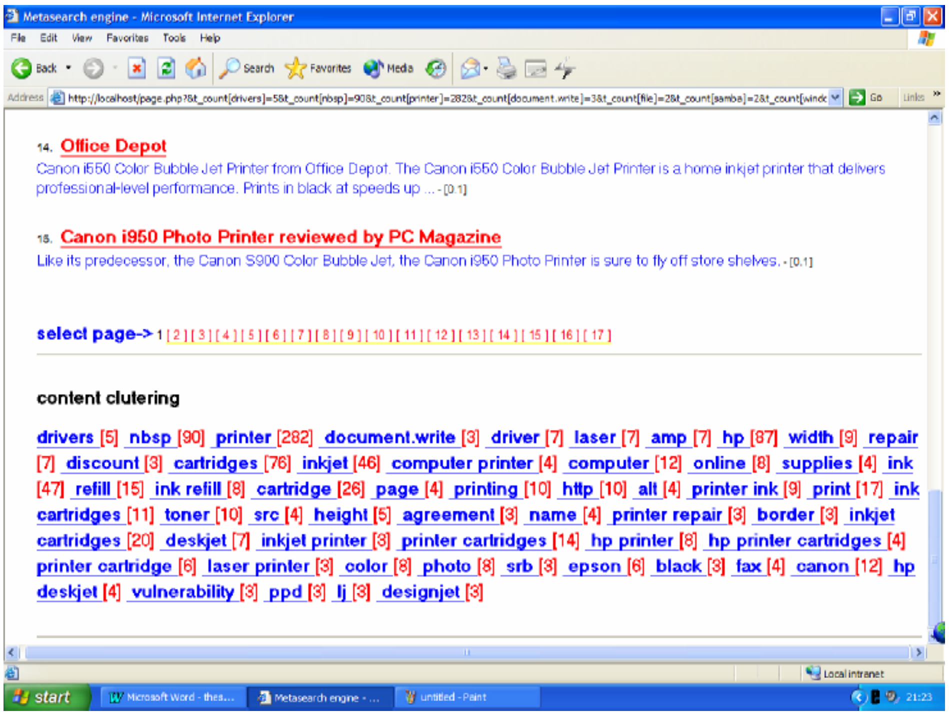
3. Inkjet Printer Cartridges
Supplies discount-priced, compatible inkjet printer cartridges and remanufactured laser toner ... BJC-600, BJ Series, Canon S400 Printer Canon inkjet printer cartridges Compaq IJ600 black, A900 ... - [0.1]

4. Canon inkjet printer cartridge and refill kit sale
... 85 StarWriter 95 StarWriter 300 StarWriter 4000 StarWriter 5000 Pro Select Printer Cartridge Number Here. Select by Canon InkJet Cartridge # **** New Replacement Cartridges ****** Canon BJ-300 Canon BJ ... - [0.1]

5. Print Pal: Canon Printer Cartridges - Canon Inkjet
... True Adventures with the King of Bluegrass - inkjet cartridge - True Balance - sun glasses True ... ... ink True Blood - gem stone - True Blue - canon printer True Devotion - credit check - True Face of ... - [0.1]

6. Inkjet refill kits, bulk inks and compatible inkje
... sg offers high quality canon inkjet printer ink supplies and canon ink refill kits for the majority of canon printers and canon fax machines. Mr. Ink Jet, Inc.sg is NOT affiliated in anyway with the ... - [0.1]

Figure 4.18 The ranked list page


14. Office Depot
Canon i550 Color Bubble Jet Printer from Office Depot. The Canon i550 Color Bubble Jet Printer is a home inkjet printer that delivers professional-level performance. Prints in black at speeds up ... - [0.1]

15. Canon i950 Photo Printer reviewed by PC Magazine
Like its predecessor, the Canon S900 Color Bubble Jet, the Canon i950 Photo Printer is sure to fly off store shelves. - [0.1]

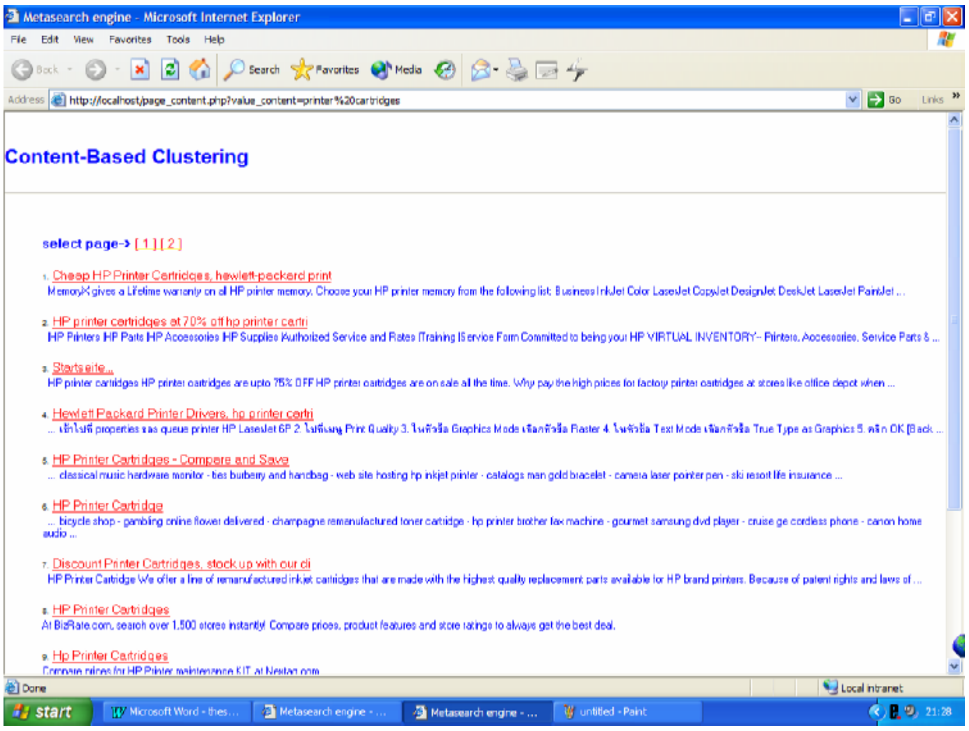
select page-> 1 [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17]

content clutering

drivers [5] nbsp [90] printer [282] document.write [3] driver [7] laser [7] amp [7] hp [87] width [9] repair [7] discount [3] cartridges [76] inkjet [46] computer printer [4] computer [12] online [8] supplies [4] ink [47] refill [15] ink refill [8] cartridge [26] page [4] printing [10] http [10] alt [4] printer ink [9] print [17] ink cartridges [11] toner [10] src [4] height [5] agreement [3] name [4] printer repair [3] border [3] inkjet cartridges [20] deskjet [7] inkjet printer [3] printer cartridges [14] hp printer [8] hp printer cartridges [4] printer cartridge [6] laser printer [3] color [8] photo [8] srb [3] epson [6] black [3] fax [4] canon [12] hp deskjet [4] vulnerability [3] ppd [3] lj [3] designjet [3]

Figure 4.19 The clustered page

The cluster member list shown in Figure 4.20 has the same information as the results in the ranked list page.


Content-Based Clustering

select page-> [1] [2]

1. Cheap HP Printer Cartridges, hewlett-packard print
MemoryX gives a Lifetime warranty on all HP printer memory. Choose your HP printer memory from the following list: Business InkJet Color LaserJet CopyJet DesignJet DeskJet LaserJet PaintJet ...

2. HP printer cartridges at 70% off hp printer cartri
HP Printers HP Parts HP Accessories HP Supplies Authorized Service and Rates Training Service Form Committed to being your HP VIRTUAL INVENTORY-- Printers, Accessories, Service Parts & ...

3. Startseite...
HP printer cartridges HP printer cartridges are upto 75% OFF HP printer cartridges are on sale all the time. Why pay the high prices for factory printer cartridges at stores like office depot when ...

4. Hewlett Packard Printer Drivers, hp printer cartri
... เข้าไปที่ properties ของ queue printer HP LaserJet 6P 2. ไปที่เมนู Print Quality 3. ในหัวข้อ Graphics Mode เลือกหัวข้อ Raster 4. ในหัวข้อ Text Mode เลือกหัวข้อ True Type as Graphics 5. คลิก OK [Back ...

5. HP Printer Cartridges - Compare and Save
... classical music hardware monitor - ties burberry and handbag - web site hosting hp inkjet printer - catalogs man gold bracelet - camera laser pointer pen - ski resort life insurance ...

6. HP Printer Cartridge
... bicycle shop - gambling online flower delivered - champagne remanufactured toner cartridge - hp printer brother fax machine - gourmet samsung dvd player - cruise ge cordless phone - canon home audio ...

7. Discount Printer Cartridges, stock up with our di
HP Printer Cartridge We offer a line of remanufactured inkjet cartridges that are made with the highest quality replacement parts available for HP brand printers. Because of patent rights and laws of ...

8. HP Printer Cartridges
At BizRate.com, search over 1,500 stores instantly! Compare prices, product features and store ratings to always get the best deal.

9. Hp Printer Cartridges
Compare prices for HP Printer maintenance KIT at Nextag.com

Figure 4.20 The cluster member list

CHAPTER 5 EXPERIMENTAL RESULTS

The chapter describes the experimental results of MEUFLACC. The objectives of experimentation can be divided into five major groups of experiments as follows:

1. The experiment of keyphrase extractions, with an objective to find the appropriate number of keyphrase extractions to use in the system.

2. The experiment of fuzzy ranking method, with an objective to evaluate the effectiveness of the system versus others search engines by using average precision and precision recall graph measurement.

3. The experiment of content-based clustering method, with an objective to evaluate cluster quality and the quality of document in clusters group. For clustering, one type of measure that evaluates effectiveness of the cluster is F-measure method.

4. The experiment of the system, with an objective to evaluate the effectiveness of the system versus Content-based Clustering Metasearch Engine(CBCMSE) [36] by using average precision, precision recall graph measurement and expected search length (ESL) [49].

5. The experiment of time response of the system, with an objective to evaluate time responding to the proposed system.

5.1 Document collection

In the experiment, the proposed system is run on a notebook computer under Windows XP Professional, PHP 4.2.0 programming language for coding the prototype and Apache 1.3.24 to be web server. The experiment involves web users assignment to evaluate the system, by dividing thirteen web users into four assignments. The first and second of assignment use five users while the third assignment uses three users for the evaluation. The fourth assignment uses the same users as the first assignment and the last assignment evaluates on the researcher. The web users who evaluate the system must have experiences in using search engine.

Web user assignments

5.1.1 Finding the appropriate number of keyphrase extractions:

The first five users are assigned to find the proper number of keyphrase extractions to be used in the system. Each participant is given two queries with a sheet describing the experiment shown in Table 5.1. After each search, the participant has to fill in a sheet the number of all clusters and the number of clusters that are satisfactory and useful. The testing numbers of keyphrase extraction are 6, 8, 10, 12, 14. All of the queries and the data could be found in appendix.

Table 5.1 An example of sheet experimenting the finding of proper number of keyphrase extraction.

Query: search engine		
Number of keyphrase extractions	Number of all clusters	Number of useful clusters
6	6	4
8	15	10
10	19	14
12	24	16
14	28	17

5.1.2 Evaluating the effectiveness of MEUFLACC versus others search engines

The second five users are assigned to evaluated the effectiveness of the approach versus others search engines by using an average precision and precision recall graph measurement. This assignment, gives each participant has to search in three types of search. The user starts with one term in searching data, then following with two terms. If search engines are MEUFLACC, each user has to define weight of query term. The value of weight are 0.1 - 1.0. Finally each user uses three terms in searching data. If search engines are MEUFLACC, each user has to define weight of query term. The value of weight are 0.1 - 1.0.

Each of the three types above, each user has to use two queries in searching eight search engines. The eight of search engines are:

1) Hotbot search engine[42].
2) Yahoo search engine[43].
3) Excite search engine[44].
4) Altavista search engine[45].
5) Google search engine[46].

6) MEUFLACC using six extracted keyphrases (MEUFLACC-6).
7) MEUFLACC using eight extracted keyphrases (MEUFLACC-8).
8) MEUFLACC using ten extracted keyphases (MEUFLACC-10).

Table 5.2 gives an example of sheet experimenting an order of relevant document. Similarity after each search, the participant fills in a sheet the order of relevant document. All of the queries and the data could be found in appendix.

Table 5.2 An example of sheet experimenting order of relevant document in one term

Search Engine: Altavista	
Query1: XML	Weight of term: -
Query2: -	Weight of term: -
Query3: -	Weight of term: -
All number of documents: 100	All relevant results: 46
Number of documents	Number of relevant documents
10	7
20	13
30	19
40	25
50	31
60	37
70	40
80	43
90	45
100	46

5.1.3 Evaluating cluster quality

The last three users are assigned to evaluate the quality of clusters and the quality of documents in clusters group by using f- measure method. This assignment, gives each participant a query with a sheet describing the experiment in Table 5.3. After each search, the user has to select four cluster groups to evaluate the quality of document in clusters group. Then, the user has to fill in a sheet the numbers of all clusters, the numbers of useful clusters, the cluster groups and an order of documents in each group. The search engines that are evaluated in this assignment are:

1) Yahoo search engine[43].
2) Open directory project (ODP)[47].
3) MEUFLACC-6.
4) MEUFLACC-8.
5) MEUFLACC-10.

All of the queries and the data could be found in appendix.

Table 5.3 An example of sheet experimenting quality of clusters

Search engine: MEUFLACC-6

Query: printer

The useful clusters: 9

All clusters: 20

Cluster group	Order of documents				
	1	2	3	4	5
Printer drivers	1	3	5	6	8
Printer port	2	5	7	10	13
Deskjet	1	3	12	15	18
Excite	3	4	8	14	16

5.1.4 Evaluating the effectiveness of MEUFLACC versus Content-based Clustering Metasearch Engine (CBCMSE)

The first five users are assigned to evaluate the effectiveness of the approach versus CBCMSE by using an average precision, precision recall graph measurement and expected search length (ESL) [49]. This assignment, gives each participant has to search the terms same use in CBCMSE. Each user has to use two queries in searching and the value of weight is 1.0.

Table 5.4 gives an example of sheet experiment an order of relevant document. Table 5.5 gives an example of sheet experiment relevant documents in the cluster and number of all of the documents in the cluster. Similarity after each search, the participant fills in a sheet the order of relevant document. All of the queries and the data could be found in appendix.

Table 5.4 An example of sheet experimenting order of relevant document

Search Engine: Altavista Query1: XML	
Number of documents	Order of relevant documents
1	1
2	3
3	4
4	8
5	15
6	18
7	20
8	26
9	31
10	35

Table 5.5 An example of sheet experimenting relevant documents in the cluster and number of all of the documents in the cluster

Used cluster number	Number of relevant documents in the cluster	Number of all of the documents in the cluster
1	5	15
2	1	2
3	1	6
4	3	8
5	6	18

5.2 Results of the experiments

5.2.1 The results of an appropriate number of keyphrase extractions

The keyphrase extraction is a powerful tool for identifying interesting content of papers. The effectiveness of the system apparently depends on the proper number of keyphrase extractions. The system therefore has to select an appropriate number of keyphrase extractions by examining with the number of all clusters and the number of useful clusters.

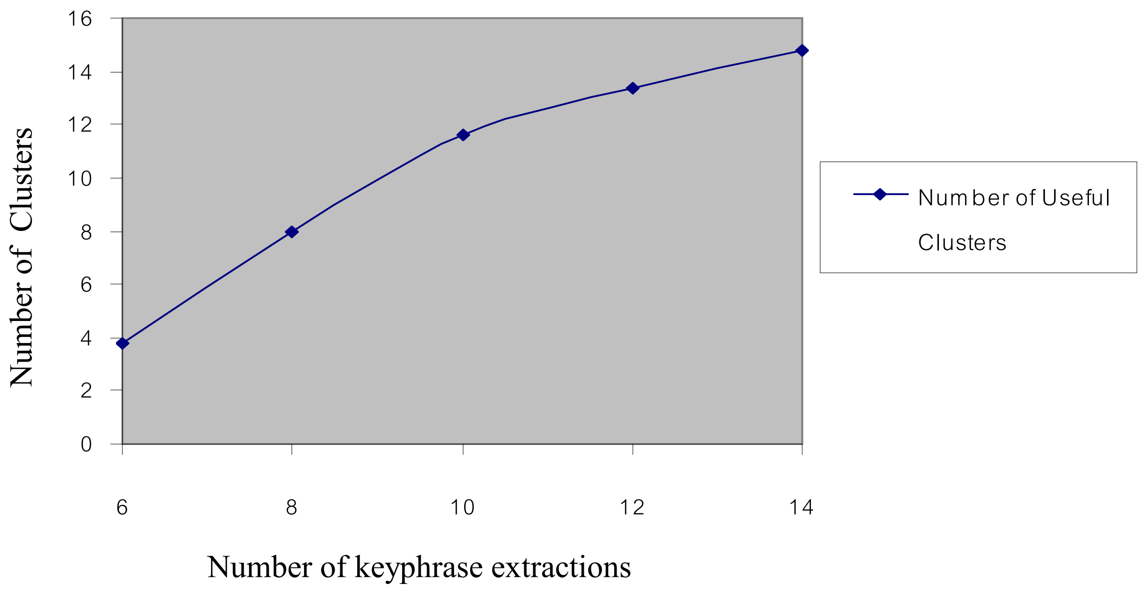


Figure 5.1 The number of useful Clusters

Figure 5.2 shows the cluster ratio of useful cluster divided by the number of all clusters.

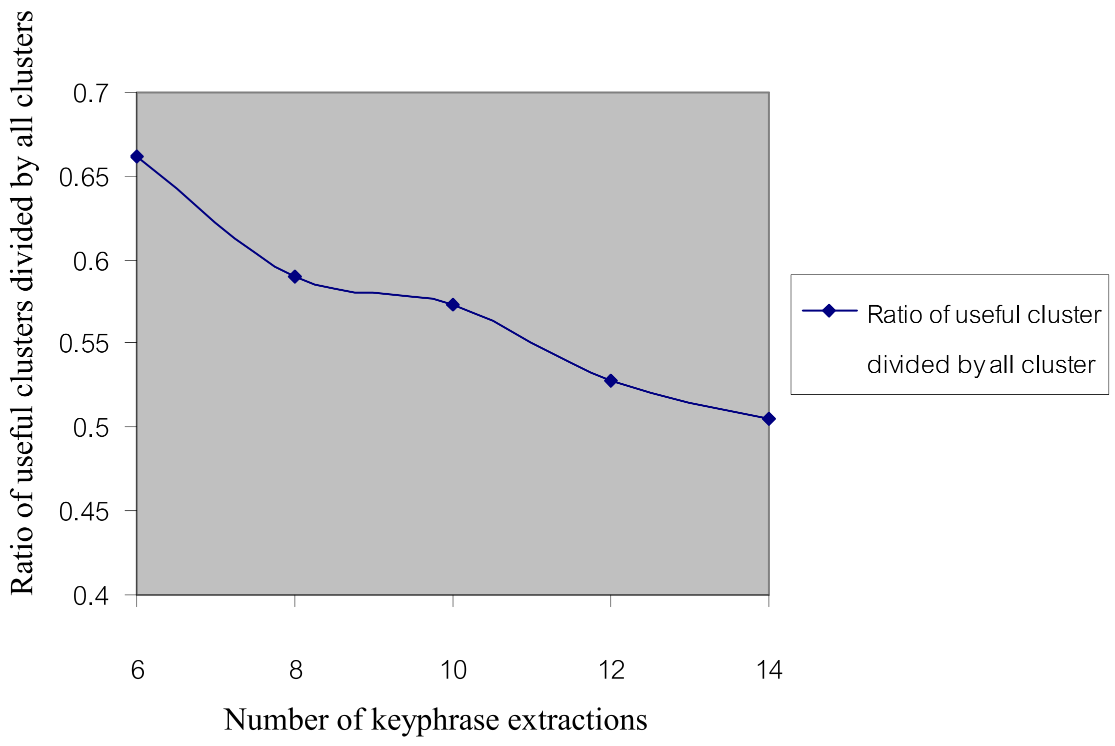


Figure 5.2 The ratio of useful clusters divided by the number of all clusters

Figure 5.1 also shows that when increasing the number of keyphrase extraction, the number of useful clusters increases. From the result, it seems that higher number of keyphrase extraction gives higher number of useful clusters. However, when comparing the result of Figure 5.1 with Figure 5.2, it is found that the ratio of useful cluster of higher number of keyphrase extraction decreases. The value of 12 and 14 keyphrase extraction are no rather than 0.55.

Since the effectiveness of the system depends on the proper number of keyphrase extraction, this thesis chooses the numbers of 6, 8 and 10 of keyphrase extraction to be the candidate for a test in next experiment.

5.2.2 The effectiveness of the approach versus others search engines

In this section, its goal is to evaluate the effectiveness of the approach and also to compare the results of the proposed system with other conventional search engines by using average precision and precision recall graph measurement.

Average precision

Average precision can be obtained from averaging the results of various queries that referred to precision versus recall. This approach is a standard evaluation strategy for information retrieval system. It is useful because it allows users to evaluate quantitatively both the quality of the results and retrieval algorithm.

As shown in Figure 5.3 below are results of average precision of one term query.

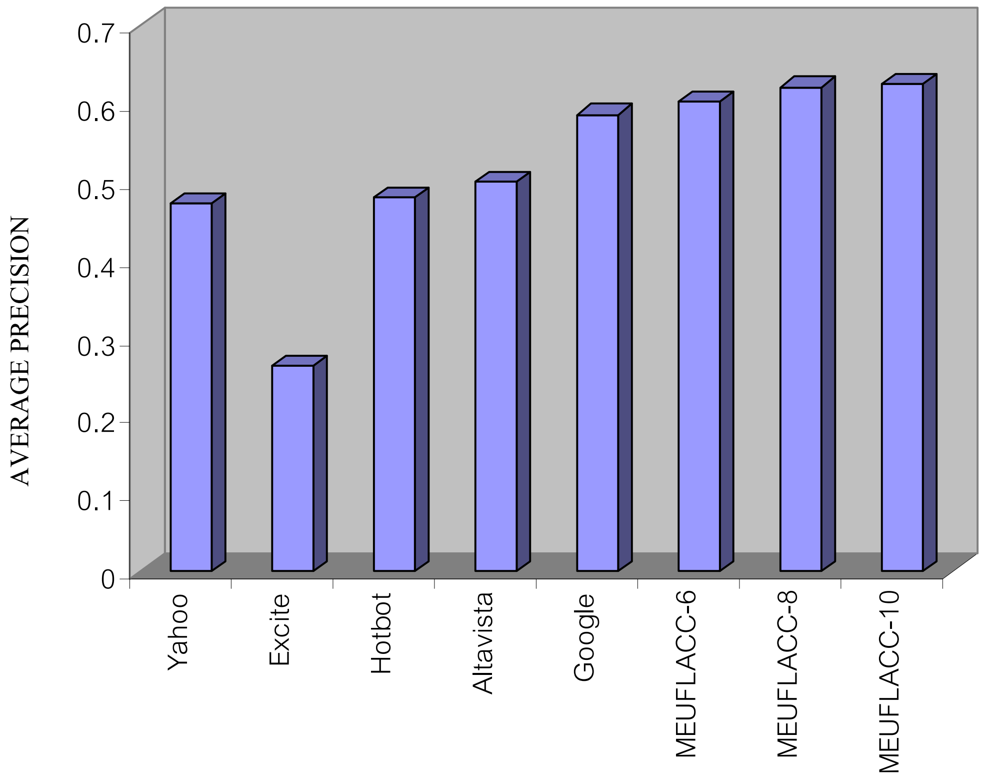


Figure 5.3 The results of average precision of one term query

From the result in Figure 5.3 above, it can be noticed that the experimental results of average precision of one term query, achieves the best effectiveness. Google search engine ranks second, Altavista ranks third, Hotbot ranks fourth, Yahoo search engine ranks fifth and Excite ranks sixth.

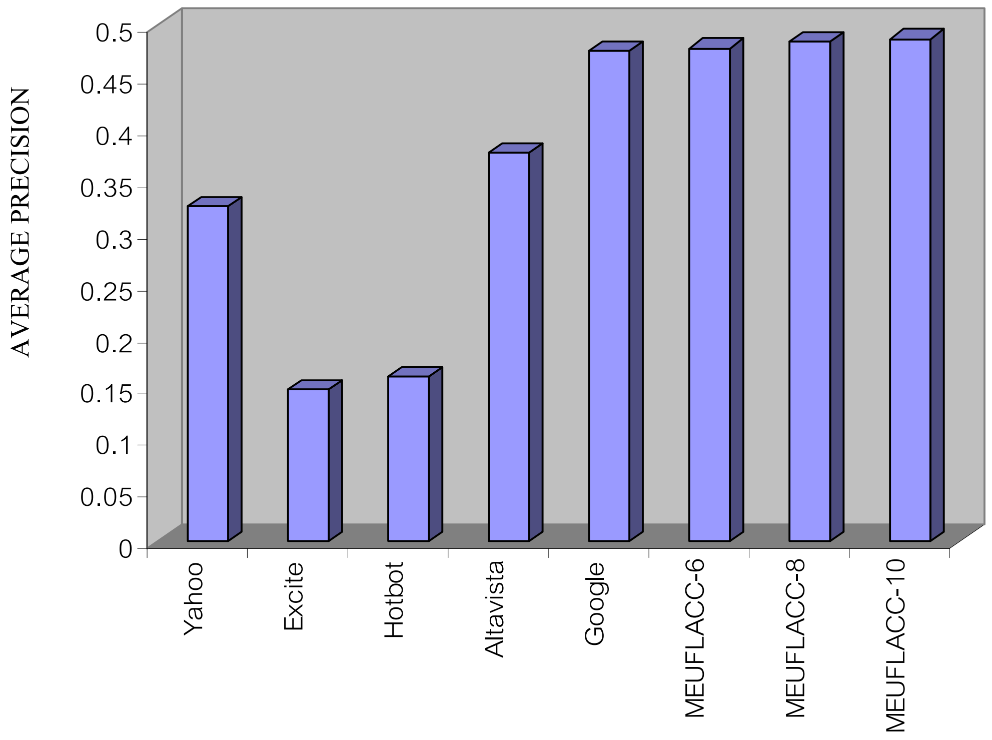


Figure 5.4 The results of average precision of two terms query

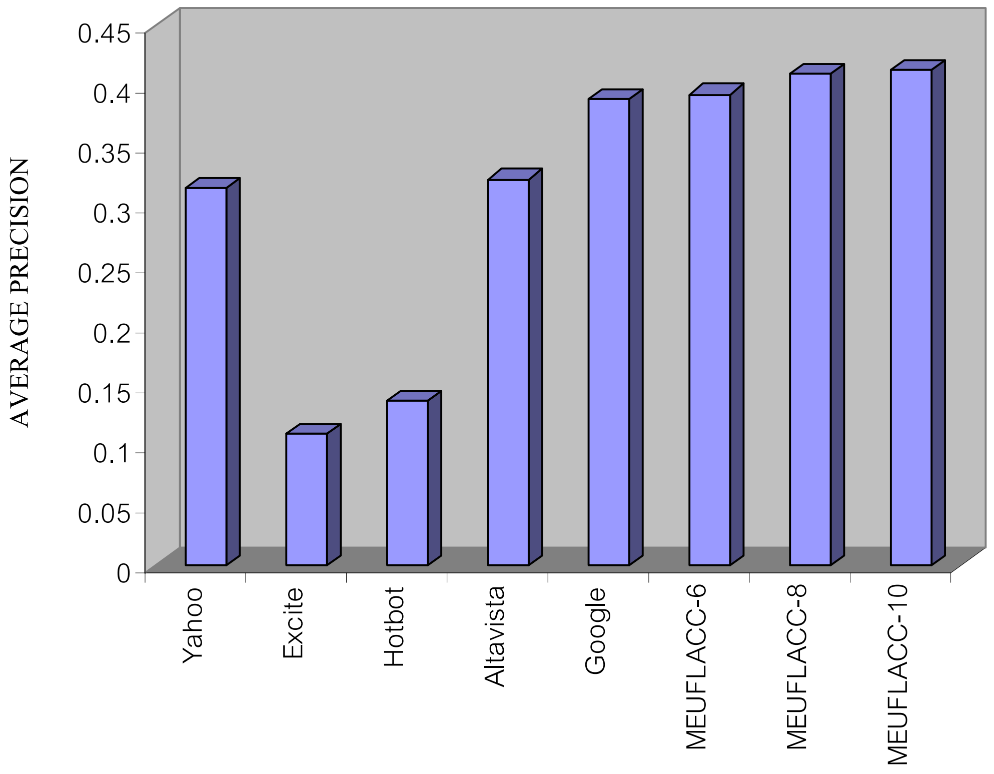


Figure 5.5 The results of average precision of three terms query

As shown in Figure 5.4, the precision values of Hotbot, and Excite search engine are less than 0.2. The precision of MEUFLACC-6 and Google search engine are equal. Further more, it is found that the proposed system also achieves the best performance. However, when comparing with the results of Figure 5.5, the value of average precision of two terms query decreases. Because the results from search engines are rarely relevant to the information needed.

According to Figure 5.5, MEUFLACC outperforms other search engines in average precision. However, MEUFLACC-6 tends to decrease when comparing the results of Figure 5.4 and Figure 5.5. The results returned from MEUFLACC-6 and Google search engine are equal. The Hotbot search engine is lower in average precision than the Altavista search engine and Yahoo search engine. The Excite search engine is the lowest on the list.

Precision-recall graph

Precision-recall graph is a series of precision and recall pairs that can be plotted on a graph. There is a measurement for counting the number of relevant documents each time, taken as increasing numbers of documents retrieved. The precision-recall graph shows the accuracy of the result of each search engine. The high value means that the system helps the web users find the information needed.

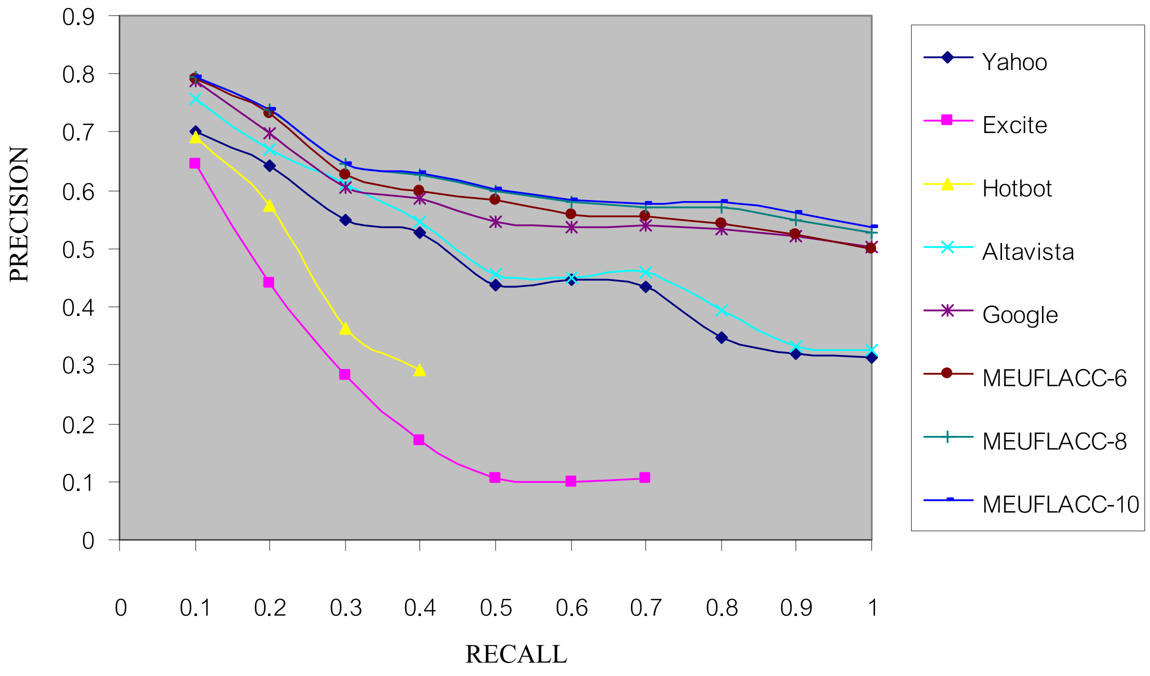


Figure 5.6 The results of precision-recall graph of one term query

According to Figure 5.6, this thesis plots the precision-recall graph. The graph shows that the proposed method ranks first on the list. The precision results of MEUFLACC are decreasing when the recall increases. The Google search engine ranks second on the list, the Altavista ranks third and the Yahoo search engine ranks fourth. The Altavista search engine is better than the Google search engine at 0.3 of recall and dropping in precision to become lower than Google search engine after that. The Excite search engine is the lowest on the list.

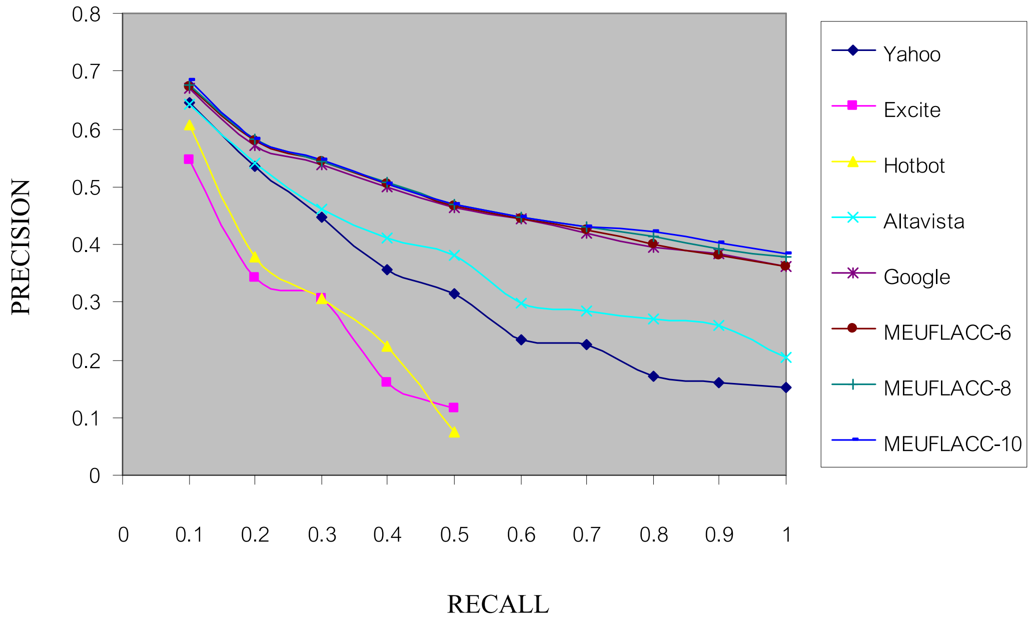


Figure 5.7 The results of precision-recall graph of two terms query

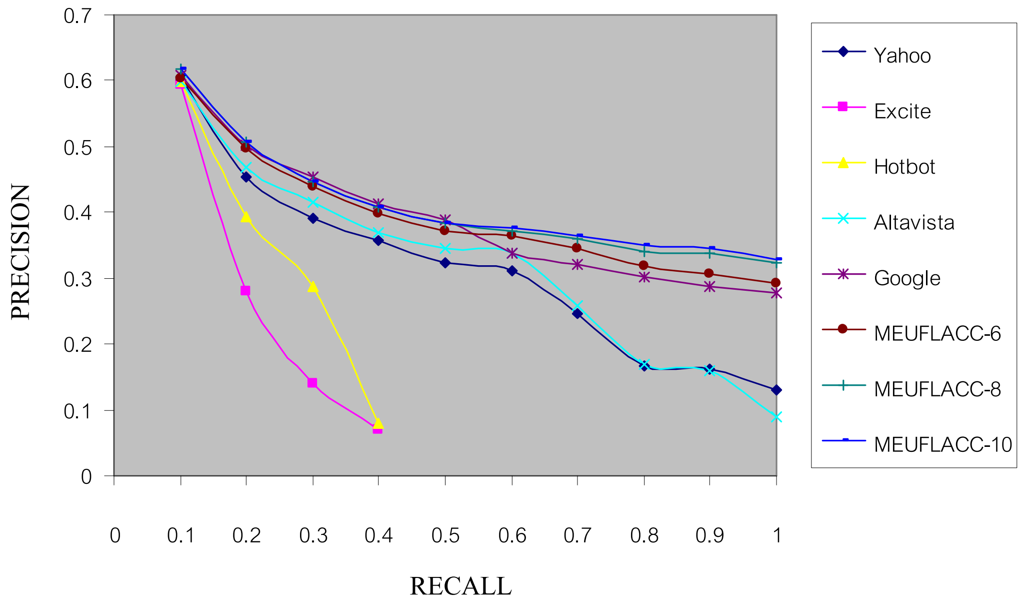


Figure 5.8 The results of precision-recall graph of three terms query

As shown in Figure 5.7, it is the result of precision-recall graph of two terms query. The proposed system also achieves the best performance. The values of precision of MEUFLACC are higher than other search engines, except at 0.5 and 0.6

of recall, the results of MEUFLACC and Google search engine are equal. The Altavista ranks third. The Yahoo search engine is equally the Altavista search engine at 0.1 and 0.2 of recall and decreasing in precision after that. The Excite search engine is the lowest on the list. The recall of Excite and Hotbot search engine has only 0.5.

As shown in Figure 5.8, the MEUFLACC ranks first rank list. But the Google has better in precision than the MEUFLACC search engine at 0.3, 0.4 and 0.5 of recall and dropping in precision to become lower than MEUFLACC search engine after that. The Altavista search engine ranks third, the Yahoo search engine ranks fourth, the Hotbot search engine ranks fifth and the Excite search engine also is the lowest. The recall of Excite and Hotbot search engine has only 0.4.

5.2.3 The evaluation of cluster quality

This part shows the results of evaluating cluster quality and the quality of document in cluster groups by using F-measure method.

F-measure

For clustering, the measure of cluster "goodness" or quality is F-measure method. The f measure has a long history, but was recently extended to document hierarchies in [48]. This measure combines the precision and recall ideas from information retrieval. The formula for F-measure is:

$$F = \frac{1}{\frac{1}{2}(1/P) + \frac{1}{2}(1/R)}$$

where:

F : the value of F-measure

P : the value of precision

R : the value of recall

Figure 5.9 shows that the F-measure of MEUFLACC-6 ranks first when comparing with other search engines. The F-measure of MEUFLACC-6 is higher than MEUFLACC-8 and MEUFLACC-10 because the precision of MEUFLACC-8 and MEUFLACC-10 is lower than MEUFLACC-6. The ODP ranks fourth, the Yahoo search engine ranks fifth.

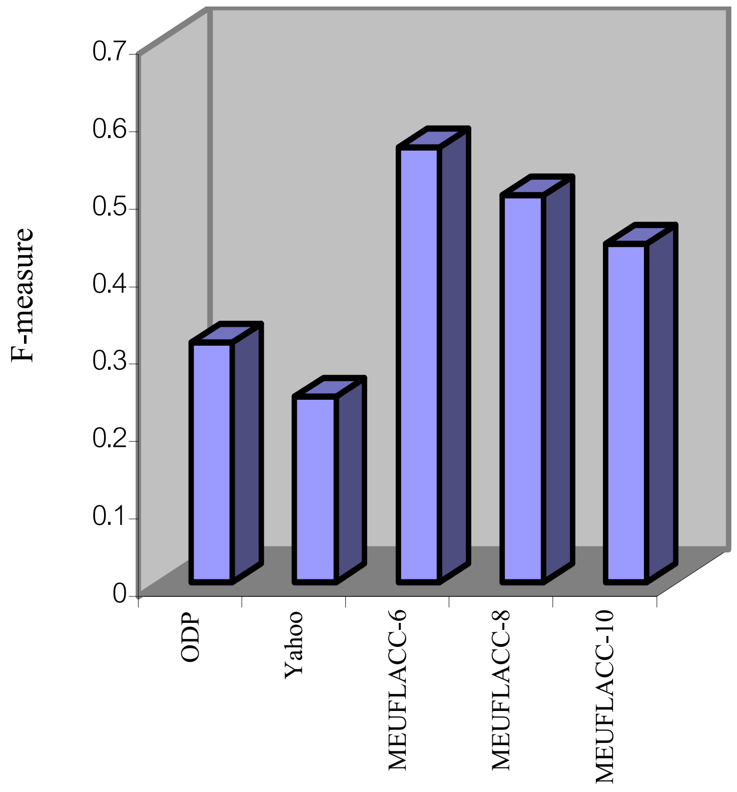


Figure 5.9 The results of F-measure for the quality of clusters

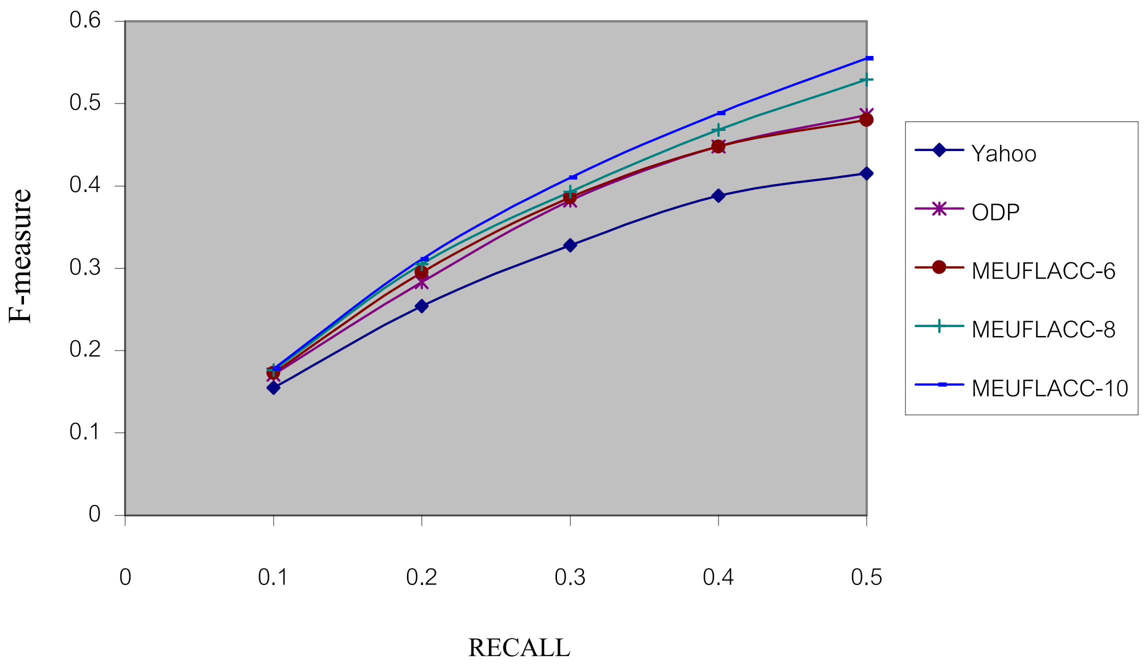


Figure 5.10 The result of the quality of document in cluster groups

According to Figure 5.10, at 0.1 of recall many search engines have equal F-measure values except the Yahoo search engine. The MEUFLACC-10 ranks first, the MEUFLACC-8 ranks second. The F-measure result of MEUFLACC-6 and ODP search engine are similar. The Yahoo search engine is lowest.

5.2.4 The effectiveness of MEUFLACC versus Content-based Clustering Metasearch Engine (CBCMSE)

In this section, its goal is to evaluate the effectiveness of the approach and also to compare the results of the proposed system with CBCMSE by using average precision, precision recall graph measurement and ESL.

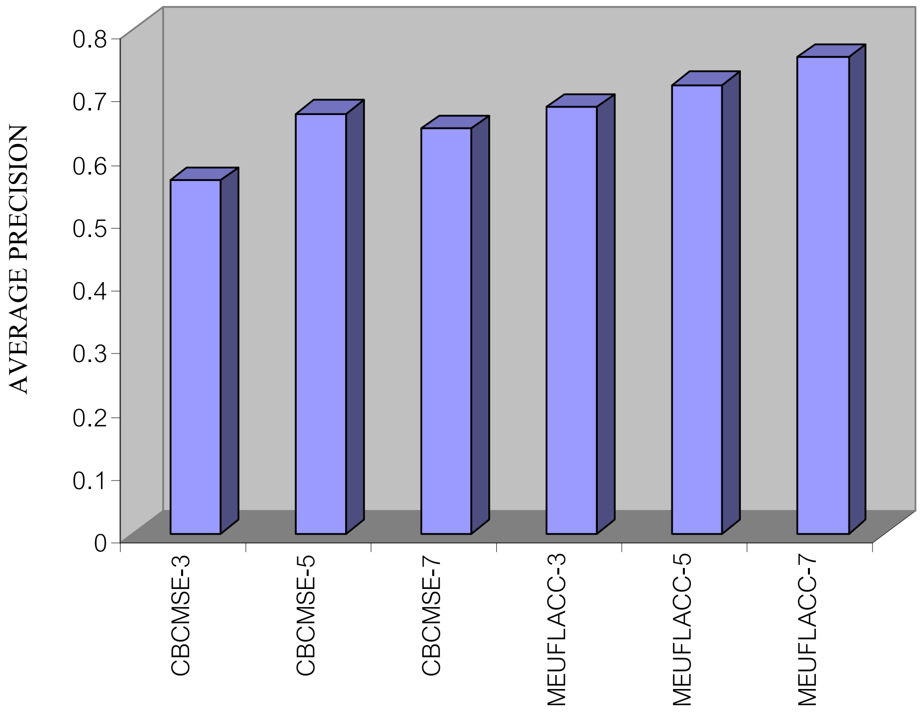


Figure 5.11 The results of average precision of MEUFLACC versus CBCMSE

From the result in Figure 5.11 above, it can be noticed that the experimental results of average precision of MEUFLACC, achieves the best effectiveness. The MEUFLACC-7 ranks first rank list, MEUFLACC-5 ranks second, MEUFLACC-3 and CBCMSE-5 is almost equally, CBCMSE-7 ranks fifth and CBCMSE-3 is the lowest.

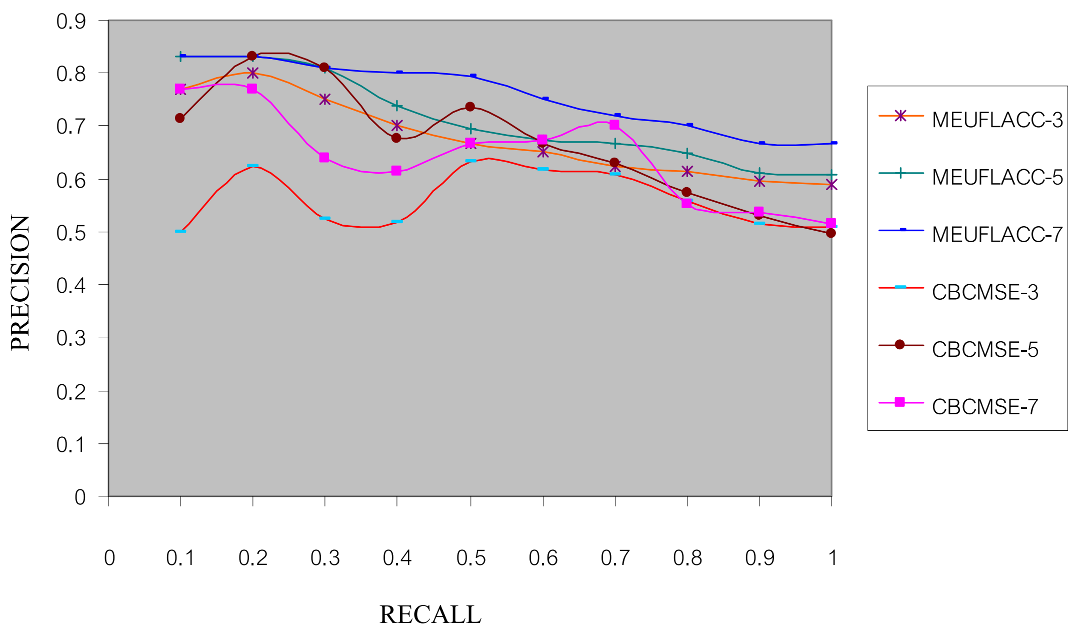


Figure 5.12 The results of precision-recall graph of MEUFLACC versus CBCMSE

As shown in Figure 5.12, the MEUFLACC-7 ranks first rank list. But the MEUFLACC-5 and MEUFLACC-3 has lower in precision than the CBCMSE-5 at 0.2, 0.3 and 0.5 of recall. The CBCMSE-7 has lower in precision than the CBCMSE-5 at 0.2, 0.3, 0.4, 0.5 and 0.8 of recall. The CBCMSE-3 is the lowest on the list.

Expected search length

In 1968, Cooper[49] stated: 'The primary function of a retrieval system is conceived to be that of saving its users to as great an extent as is possible, the labour of perusing and discarding irrelevant documents, in their search for relevant ones'. ESL measures the number of irrelevant documents would have to read through before finding a specified number of relevant items. The lower value means that the system provides higher convenience for the users.

$$ESL(q) = j + \frac{n * s}{r + 1}$$

where

j : the total number of documents non-relevant to q in all levels preceding the final

n : the number of non-relevant documents in the final level

r : the number of relevant documents in the final level

s : the number of relevant documents required from the final level

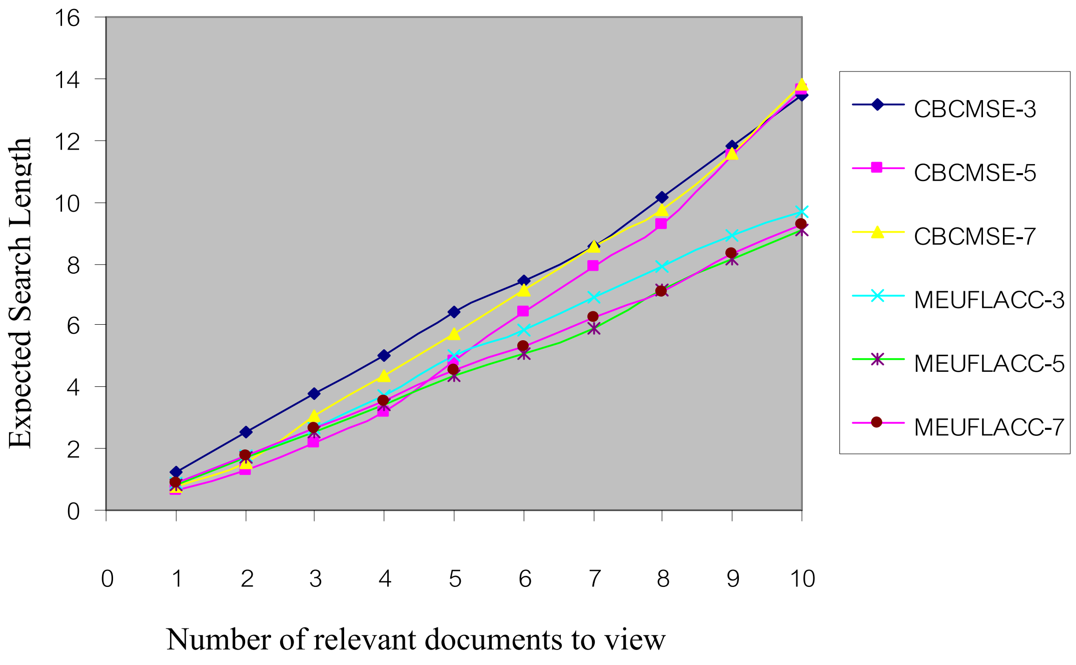


Figure 5.13 ESL graph of all engines

This assignment uses data in the appendix to calculate the ESL. The Figure 5.13 shows ten number-to-view comparing to the ESL in the graph within ten relevant documents how many irrelevant documents are found so far.

As shown in Figure 5.13, the result shows that the MEUFLACC is help web users find relevant documents faster than CBCMSE. But the MEUFLACC-3, MEUFLACC-5 and MEUFLACC-7 has lower in ESL than CBCMSE-5 at 1, 2, 3 and 4 of number of relevant documents to view. MEUFLACC-5 and MEUFLACC-7 is almost equally. The CBCMSE-3 is the highest on the list.

5.2.5 The response time of the system

The metasearch engine using fuzzy logic and content-based clustering has two processes to connect with the targeted search engines and URL site via Internet to retrieve data. The first process takes several minutes to obtain HTML pages that are returned from the targeted search engines. Another process uses the parsed results which are the URL' s to request the HTML from each resultant site. This process takes longer time than the other process because it waits until all sites return the HTML to the system. Table 5.6 below shows response time of the system which depends on speed of Internet and numbers of documents.

Table 5.6 The response time of the system.

Speed internet	Numbers of document	Response time of new query / minute	Response time of same query/ minute	Ratio of new query per same query
36 Kbps	25	≈ 27	≈ 8	3.375
	50	≈ 60	≈ 15	4
	100	≈ 120	≈ 25	4.8
	200	≈ 200	≈ 38	5.263
	500	≈ 480	≈ 55	8.727
45 Kbps	25	≈ 18	≈ 5	3.6
	50	≈ 45	≈ 8	5.625
	100	≈ 90	≈ 15	6
	200	≈ 150	≈ 25	6
	500	≈ 350	≈ 40	8.75
56 Kbps	25	≈ 16	≈ 5	3.2
	50	≈ 40	≈ 8	5
	100	≈ 75	≈ 13	5.769
	200	≈ 135	≈ 20	6.75
	500	≈ 300	≈ 35	8.571

Table 5.6 reveals that, the proposed system has limitation on retrieval time. The retrieval time is the weakness of the system because the system takes several minutes to obtain the documents. The main reason of the delay is the time spending on retrieving actual pages. This thesis therefore improves the efficiency of the system by keeping the documents in the database when searching. It can reduce time spend on searching when web users use same query.

CHAPTER 6
DISCUSSION AND CONCLUSION

This chapter attempts to show the advantages and the disadvantages of the proposed system by analyzing and discussing the results of the experiments presented in the previous chapter. Some experiments in the previous chapter are performed to measure the effectiveness of the MEUFLACC. Also a discussion on the limitation of the search engines and a suggestion for future research are proposed.

6.1 Discussion

It is the intention of the thesis to improve the weakness of conventional search engines and to provide some features for helping users in finding relevant information that meets their needs. These prominent features of MEUFLACC are:

Ranking technique

The conventional search engines use very simple ranking algorithm, which causes scattered presentation of relevant documents in their results. Being one of the prominent feature of the metasearch engine, the ranking technique should be capable of retrieving relevant documents according to their weighting.

According to Figure 5.3, Figure 5.4 and Figure 5.5 it is found that the proposed system achieves the best performance. However, when comparing the average precision result of Figure 5.5 with the average precision results of Figure 5.3 and Figure 5.4, the value of average precision of three terms query is lower than two terms query and one term query. The average precision result of one term query is regarded as the best, because the results from search engines are rarely relevant to the information users need.

Clustering algorithm

The clustering algorithm of the system provides proper groups that relate documents together. The documents may be related in any manner. It has been shown that post-retrieval documents clustering produce superior results to pre-retrieval clustering. This is due to the fact that clusters are computed based on the returned document set; the cluster boundaries are drawn to appropriately partition the set of documents at hand. In contrast, pre-retrieval clusters might be based on features that are infrequent in the cluster formation. For this reason, the metasearch engine uses the post clustering technique instead, as already described in the previous chapter.

According to Figure 5.9, the f-measure of MEUFLACC-6 ranks first when compared with others search engine. The f-measure of MEUFLACC-6 is higher than MEUFLACC-8 and MEUFLACC-10 because the precision of MEUFLACC-8 and MEUFLACC-10 are lower than MEUFLACC-6. The ODP ranks fourth, the Yahoo search engine ranks fifth.

Maximum coverage

Due to the tremendous growth in data in recent years, it is impossible for a single search engine to index all the web documents. And the main challenge faced today by the internet information retrieval systems is the incomplete search. Therefore the proposed system is designed in such a way that it is able to provide maximum coverage.

Integration of document presentation

The aim of the thesis is to help users find relevant results with higher precision than other search engine and make the retrieved results easy to browse by pulling results of the same content into the same group. The search results can be greatly improved by integrating the fuzzy ranking and content-based clustering techniques. It is the mechanism of the presentation.

Reduce retrieval time

The thesis also improves the efficiency of the system by keeping the documents in the database when searching. It can reduce time for searching when web users request for the same query. The system will use the information in the database, instead of asking the original server again.

According to Table 5.5 which shows the ratio of new query to same query, it takes less time for the client to receive the results and display it when using the same query.

6.2 Conclusions

The conclusions of the research are:

1. The number of keyphrase extractions has an influence on the effectiveness of the proposed system. The system will provides high precision and recall when using the proper number of keyphrase extractions.
2. Retrieval speed in an environment with larger size of documents collection takes more time.
3. Reducing time in searching by MEUFLACC when web users request for the same query as it will use the information in the database, instead of asking the original server for it again.

4. Reducing the noise by extracting only contents of the documents rather than all words appear in documents.
5. Defining weight of query will help users find relevant documents which are close to their interest.
6. Content-based clustering is aimed to help users find relevant results when the users are unfamiliar with the topics that they are looking for, because they are novices at performing searches.
7. Decreasing the number of keyphrase extractions would decrease the effectiveness of the system when web users use a three-term query.
8. The results of image search will be wrong when users use the same image but different size or date.
9. The limitation of MEUFLACC is the retrieval time. It is regarded as a weakness of this system because it takes several minutes to obtain the documents. The main reason is the delay the time spent on retrieving actual pages.

6.3 Future work

Finally, it is the purpose of the thesis to give a guideline for some interesting directions for future works as follows.

1. Computer clustering: this approach is connecting two or more computers together in such a way that they behave like a single computer. Computer Clustering is used for parallel processing, load balancing and fault tolerance.

2. Connect via proxy: this system can not connect via a proxy. However web developers can improve MEUFLACC by updating the program in a connection module. This method reduces latency because it takes less time for the client to receive the results and display them. The traffic is avoidable by reducing the amount of bandwidth used by a client.

3. Term weighting: KEA does not support THAI language, however web developers can improve this by finding other keyphrase extraction that support THAI language.

4. Exploring other areas of information retrieval with an alternative and probably better approach can be investigated and applied.

REFERENCES

1. Netsizer [Online], Available from: http://www.netsizer.com [Accessed 2002].
2. Kehoe, C. and Pitkow, J. Tenth WWW Survey Report. The Graphics, Visualization & Usability Center: Georgia Institute of Technology, 1999.
3. D.SullivanSearch EngineWatch: Mainpage. http://www.searchenginewatch.com, 1997.
4. S. Lawrence and C. L. Giles. Searching the World Wide Web (in reports). Science, 280(5360), 1998.
5. Y. Wang, M. Kitsuregawa. Link-based Clustering of Web Search Results. In Proceedings of The Second International Conference on Web-Age Information Management(WAIM2001), 2001.
6. G. Bordogna and G. Pasi. A Fuzzy Rule Based Mechanism for Ranking Multiple Information Sources. Institute of Technology Information Multimedia, Milan, Italy, 1999.
7. Robert R. Korfhage. Information Storageand Retrieval. New York: John Wiley&Son, Inc., 1997.
8. Metro, H., Infoseek CEO. CNBC, 1998.
9. Ian H. Witten, Gordon W. Paynter E. Frank, C. Gutwin and Craig G. Nevill-Manning. KEA: Practival Automatic Keyphrase Extraction. Department of Computerscience, University of Waikato, New Zealand, 1999.
10. Lancaster, F.W., Information Retrieval Systems: Characteristics, Testing and Evaluation. Wiley, 1968.
11. Maxim Lifantsev. A System for Collaborative Web Resource Categorization and Ranking. Department of Computer Science, SUNY at Stony Brook, New York, 2001.
12. Baeza-yates,R., Ribeiro-Neto, B. Modern Information Retrieval. ACM Press, 1999.

13. Mei Kobayashi and Koichi Takeda, Information Retrieval on the Web. ACM Computing Surveys, Vol. 32, No. 2, 2000.
14. George J. Klir, Ute H. ST.Clair, Bo Yuan. Fuzzy Set Theory. Prentice-Hall,Inc., 1997.
15. Lecture 3: Extended IR Models. Johan Bollen [Online] Available from http:// www.cs.odu.edu/~jbollen/spring03_IR/cs695_lect3.pdf[Accessed 2003].
16. Jose A. Olivas, Pablo J. Garces, Grancisco P. Romero. FISS: Application of Fuzzy Technologies to an Internet Metasearcher. Department of Computer of Science, UCLM, Spain, 2000.
17. D. Lucarella and R. Morara. FIRST: Fuzzy information retrieval system. Journal of Information Science, vol. 17, no. 1, pp. 81-91, 1991.
18. Dwi H.Widyantoro, John Yen. A Fuzzy Ontology-based Abstract Search Engine and Its user studies. Department of Computer of Science, Texas A&M University, USA, 2001.
19. Masoud Nikravesh. Present Fuzzy Conceptual-Based Search Engine Using Conceptual Semantic Indexing. Department of Computer of Science, University of California at Berkley, USA, 2002.
20. C. J. van RIJSBERGEN. Information Retrieval. Department of Computing Science, University of Glasgow, 2000.
21. Search Engine Basics [Online] Available from http://www.trinityvalleyschool.org/ academics/library/search.html [Accessed 2002].
22. J. Cho and H. Garcia-Molina. The Evolution of the Web and Implications for an Incremental Crawler. Department of Computer Science, Stanford, 2000.
23. Understanding Search Engines and Directories [Online] Available from http://www.ils.unc.edu/fents/ed/ [Accessed 2002].
24. W. Meng, K. Liu, C. Yu, W. Wu and N. Rishe. Estimating the Usefulness of Search Engines. Department of Computer Science, SUNY at Binghamton, New York, 1999.
25. Patrick Pantel, Dekang Lin. Document Clustering with Committees: University of Alberta, 2002.

26. Ics [Online] Available from http://www.ics.uci.edu/~eppstein/280/cluster.html [Accessed 2002].

27. Promoim [Online] Available from http://www.promoim.co.kr/ Education/Expert /Cluster%20anaysis.ppt [Accessed 2002].

28. Jones, S. and Paynter, G.W. Human Evaluation of Kea, an Automatic Keyphrasing Sysem. ACM Press, pp. 148-156, 2001.

29. An Evaluation of Document Keyphrase Sets [Online] Available from http://jodi.ecs.soton.ac.uk/Articles/v0/i01/Jones/#Turney2000 [Accessed 2002].

30. KEA:Automatic Keyphrase Extraction [Online] Available from http://www.nzdl.org/Kea/ [Accessed 2002].

31. Frank, E., Paynter, G., Witten, I., Gutwin, C. and Nevill-Manning, C. Domain-specific ACM Press, pp. 668-673, 1999.

32. Turney, P.D. Learning Algorithms for Keyphrase Extraction. Information Retrieval, 2(4), 303-336, 2000.

33. Parinya Saeang. Automatic hypertext generation by application of lexical chain. Mahidol University, 2002.

34. Extractor: Text Summarization Software. [Online] Available from http://extractor.iit.nrc.ca [Accessed 2004].

35. Search Engine Statistics [Online] Available from http://www.searchengineshow down.com/stats/size.shtml [Accessed 2002].

36. Tanasai Sucontphunt. Content-Based Clustering Metasearch Engine. Mahidol University, 2000.

37. William B. Frakes and Ricardo Baeza-Yates. Information Retrieval Data Structures & Algorithm. New Jersey: Prentice Hall; 1992.

38. Kucera, H. and Francis, W.N. Computational analysis of present-day American English. Brown University Press, Providence, 1967.

39. Lovins, J.B. Development of a stemming algorithm. Mechanical Translation and Computational Linguistics, 11, 22-31, 1968.

40. Fayyad, U.M. and Irani, K.B. Multi-interval discretization of continuous-valued attributes for classification learning. *Proc IJCAI'93*, 1022-1027, 1993.

41. Domingos, P. and Pazzani, M. On the optimality of the simple bayesian classifier under zero-one loss. *Machine Learning*, 29 (2/3), 103-130, 1997.
42. Hotbot Search Engine [Online] Available from http://www.hotbot.com [Accessed 2003].
43. Yahoo Search Engine [Online] Available from http://www.yahoo.com [Accessed 2003].
44. Excite Search Engine [Online] Available from http://www.excite.com [Accessed 2003].
45. Altavista Search Engine [Online] Available from http://www.altavista.com [Accessed 2003].
46. Google Search Engine [Online] Available from http://www.google.com [Accessed 2002].
47. Open Directory Project [Online] Available from http://www.omoz.org [Accessed 2002].
48. M. Steinbach, G. Karypis and V. Kumar. A Comparison of Document Clustering Techniques. Department of Computer Science and Engineering, University of Minnesota, 2000.
49. Cooper W. S. Getting beyond Boole. Information Processing and Management 1988; 24(3):243-248.

APPENDIX

APPENDIX

Number of Keypharse Extraction	Number of Cluster	Queries:				
		Fuzzy logic	Search engine	PHP	Printer	Scanner
6	All clusters	4	6	7	7	8
	Useful Clusters	3	4	4	5	4
8	All clusters	13	15	15	17	14
	Useful Clusters	9	10	7	11	6
10	All clusters	18	19	18	20	17
	Useful Clusters	12	14	9	13	12
12	All clusters	20	24	26	27	24
	Useful Clusters	14	16	10	13	14
14	All clusters	27	28	30	29	28
	Useful Clusters	15	17	12	15	15

Number of Keypharse Extraction	Number of Cluster	Queries:				
		Virus Computer	PDA	Harddisk	CGI	Windows
6	All clusters	5	9	4	6	6
	Useful Clusters	4	5	3	4	5
8	All clusters	13	16	17	16	20
	Useful Clusters	9	8	10	10	12
10	All clusters	23	20	22	22	25
	Useful Clusters	11	11	10	12	13
12	All clusters	26	24	28	25	30
	Useful Clusters	14	11	14	13	15
14	All clusters	29	28	32	27	35
	Useful Clusters	15	13	15	13	18

Query No. 1: Kasembundit

Search Engines	Number of relevant documents									
	11	22	33	44	55	66	77	88	99	110
Google	8	17	25	30	36	45	55	63	69	72
Altavista	7	15	24	27	36	41	50			
Hotbot	7	15	21							
Excite	8	12	19	21						
Yahoo	7	14	22	27	37	42	48			
MEUFLACC – 6	8	18	27	33	38	48	57	63	68	73
MEUFLACC – 8	8	18	27	35	41	50	58	65	71	76
MEUFLACC – 10	8	18	27	36	42	50	58	67	73	78

Query No. 2: Loikrathong

Search Engines	Number of relevant documents									
	53	106	159	212	265	318	371	424	477	530
Google	35	72	90	131	154	174	194	225	239	252
Altavista	30	65	78	117						
Hotbot	31									
Excite	30									
Yahoo	28	62	72	112						
MEUFLACC – 6	37	78	92	141	160	173	189	223	238	250
MEUFLACC – 8	37	78	92	146	163	175	190	227	243	254
MEUFLACC – 10	37	78	92	147	167	179	195	232	248	259

Query No. 3: Nangtalung

Search Engines	Number of relevant documents									
	3	6	9	12	15	18	21	24	27	30
Google	3	4	6	8	9	11	13	15	15	17
Altavista	3	5	6	7	8	10	11	13	13	15
Hotbot	2	4	5	7						
Excite	2	3	5	6	7	9	10			
Yahoo	2	4	6	7	8	9	10	10	12	14
MEUFLACC – 6	3	5	6	8	10	12	13	15	16	17
MEUFLACC – 8	3	5	7	9	10	13	14	16	17	18
MEUFLACC – 10	3	5	7	9	10	13	15	17	18	19

Query No. 4: Chartpattana

Search Engines	Number of relevant documents									
	26	52	78	104	130	156	182	208	234	260
Google	21	40	56	68	81	98	108	123	140	146
Altavista	19	37	50	62	72	84	97	111	130	143
Hotbot	18	32	41	49						
Excite	18	33	37							
Yahoo	17	34	44	58	65	78	90	102	124	136
MEUFLACC – 6	21	41	55	67	81	96	109	121	139	144
MEUFLACC – 8	21	42	57	69	84	101	113	127	142	156
MEUFLACC – 10	21	42	57	69	84	101	113	127	146	159

Query No. 5: Kwanruen

Search Engines	Number of relevant documents									
	39	78	117	156	195	234	273	312	351	390
Google	32	54	72	89	101	126	144	165	180	198
Altavista	34	51	68	76	92	114	136	155		
Hotbot	28	43	61	67						
Excite	32	43								
Yahoo	30	49	64	72	83	108	125	142		
MEUFLACC – 6	32	55	78	90	103	123	143	160	181	195
MEUFLACC – 8	33	57	80	92	107	130	149	163	185	198
MEUFLACC – 10	33	57	80	92	107	130	149	163	187	201

Query No. 6: Baanlaesuan

Search Engines	Number of relevant documents									
	37	74	111	148	185	222	259	296	333	370
Google	30	54	71	87	102	121	144	157	162	181
Altavista	31	50	67	81	95	113	137	143	152	168
Hotbot	32	52								
Excite	32	41								
Yahoo	30	47	62	76	86	104	132	135	147	161
MEUFLACC – 6	30	55	73	88	103	120	140	161	169	178
MEUFLACC – 8	30	56	73	89	105	125	145	167	175	182
MEUFLACC – 10	30	56	73	89	105	125	145	168	177	184

Query No. 7: Songkarn

Search Engines	Number of relevant documents									
	110	220	330	440	550	660	770	880	990	1100
Google	72	117	142	181	236	266	313	347	382	420
Altavista	62	102	134	171	214	251	299	325	367	396
Hotbot	59	110	119	161						
Excite	41									
Yahoo	67	112	124	167	202	243	284	305	355	384
MEUFLACC – 6	74	118	151	185	244	270	323	351	390	424
MEUFLACC – 8	74	118	151	186	247	281	336	363	397	432
MEUFLACC – 10	74	118	151	186	247	281	336	363	397	432

Query No. 8: Suvarnabhumi

Search Engines	Number of relevant documents									
	260	520	780	1040	1300	1560	1820	2080	2340	2600
Google	191	277	320	386	451	504	563	663	734	815
Altavista	179	260	301	372	423	482	541	623	722	788
Hotbot	164	227	276	332						
Excite	42									
Yahoo	172	254	291	356	411	467	523	603	698	755
MEUFLACC – 6	188	280	332	379	460	501	557	634	691	761
MEUFLACC – 8	188	280	332	380	470	511	568	651	701	771
MEUFLACC – 10	188	280	332	380	470	511	568	651	712	778

Query No. 9: Northbangkok

Search Engines	Number of relevant documents									
	12	24	36	48	60	72	84	96	108	120
Google	8	15	21	27	33	42	50	58	67	71
Altavista	8	14	22	28	31	44	51	57	65	71
Hotbot	7	14								
Excite	7	15	20							
Yahoo	8	15	22	26	30	41	48	52	62	68
MEUFLACC – 6	8	15	23	27	34	41	50	57	66	72
MEUFLACC – 8	8	15	24	31	36	43	50	58	67	75
MEUFLACC – 10	8	15	24	31	36	43	50	60	69	77

Query No. 10: Nakornphanam

Search Engines	Number of relevant documents									
	1	2	3	4	5	6	7	8	9	10
Google	1	2	2	3	3	3	4	4	5	5
Altavista	1	2	3	3	3	3	4	5	5	5
Hotbot	1	2	2	3						
Excite	1	2	2	3	3	3	4			
Yahoo	1	2	2	3	3	4	4	4	5	5
MEUFLACC – 6	1	2	2	3	4	4	5	5	5	5
MEUFLACC – 8	1	2	2	3	4	4	5	6	6	6
MEUFLACC – 10	1	2	2	3	4	4	5	6	6	6

Search Engine: Hotbot **Queries: one terms**

Recall	Precision										
	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Average
0.1	0.636	0.585	0.667	0.692	0.718	0.865	0.536	0.631	0.583	1.000	0.691
0.2	0.682		0.667	0.615	0.551	0.703	0.500	0.437	0.583	1.000	0.574
0.3	0.636		0.556	0.526	0.521		0.361	0.354		0.667	0.362
0.4			0.583	0.471	0.429		0.366	0.319		0.750	0.292
0.5	0.636	0.585	0.667	0.692	0.718	0.865	0.536	0.631	0.583	1.000	0.691
0.6											
0.7											
0.8											
0.9											
1.0											
Average Precision of Hotbot											0.479

Search Engine: Yahoo **Queries: one terms**

Recall	Precision										
	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Average
0.1	0.636	0.528	0.667	0.654	0.769	0.811	0.609	0.662	0.667	1.000	0.700
0.2	0.636	0.585	0.667	0.654	0.628	0.635	0.509	0.488	0.625	1.000	0.643
0.3	0.667	0.453	0.667	0.564	0.547	0.559	0.376	0.373	0.611	0.667	0.548
0.4	0.614	0.528	0.583	0.558	0.462	0.514	0.380	0.342	0.542	0.750	0.527
0.5	0.673		0.533	0.500	0.426	0.465	0.367	0.316	0.500	0.600	0.438
0.6	0.636		0.500	0.500	0.462	0.468	0.368	0.299	0.569	0.667	0.447
0.7	0.623		0.476	0.495	0.458	0.510	0.369	0.287	0.571	0.571	0.436
0.8			0.417	0.490	0.455	0.456	0.347	0.290	0.542	0.500	0.350
0.9			0.444	0.530		0.441	0.359	0.298	0.574	0.556	0.320
1.0			0.467	0.523		0.435	0.349	0.290	0.567	0.500	0.313
Average Precision of Yahoo											0.472

Note: Q are the queries

Q1= Kasembundit	Q2= Loikrathong	Q3= Nangtalung
Q4= Chartpattana	Q5= Kwanruen	Q6= Baanlaesuan
Q7= Songkarn	Q8= Suvarnabhumi	Q9= Northbangkok
Q10=Nakornphanam		

Search Engine: Excite **Queries: one terms**

Recall	Precision										
	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Average
0.1	0.727	0.566	0.667	0.692	0.821	0.865	0.373	0.162	0.583	1.000	0.646
0.2	0.545		0.500	0.635	0.551	0.554			0.625	1.000	0.441
0.3	0.576		0.556	0.474					0.556	0.667	0.283
0.4	0.477		0.500							0.750	0.173
0.5			0.467							0.600	0.107
0.6			0.500							0.500	0.100
0.7			0.476							0.571	0.105
0.8											
0.9											
1.0											
Average Precision of Excite											0.264

Search Engine: Altavista **Queries: one terms**

Recall	Precision										
	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Average
0.1	0.636	0.566	1.000	0.731	0.872	0.838	0.564	0.688	0.667	1.000	0.756
0.2	0.682	0.613	0.833	0.712	0.654	0.676	0.464	0.500	0.583	1.000	0.672
0.3	0.727	0.491	0.667	0.641	0.581	0.604	0.406	0.386	0.611	1.000	0.611
0.4	0.614	0.552	0.583	0.596	0.487	0.547	0.389	0.358	0.583	0.750	0.546
0.5	0.655		0.533	0.554	0.472	0.514	0.389	0.325	0.517	0.600	0.456
0.6	0.621		0.556	0.538	0.487	0.509	0.380	0.309	0.611	0.500	0.451
0.7	0.649		0.524	0.533	0.498	0.529	0.388	0.297	0.607	0.571	0.460
0.8			0.542	0.534	0.497	0.483	0.369	0.300	0.594	0.625	0.394
0.9			0.481	0.556		0.456	0.371	0.309	0.602	0.556	0.333
1.0			0.500	0.550		0.454	0.360	0.303	0.592	0.500	0.326
Average Precision of Altavista											0.5

Note: Q are the queries

Q1= Kasembundit	Q2= Loikrathong	Q3= Nangtalung
Q4= Chartpattana	Q5= Kwanruen	Q6= Baanlaesuan
Q7= Songkarn	Q8= Suvarnabhumi	Q9= Northbangkok
Q10=Nakornphanam		

Search Engine: Google **Queries: one terms**

Recall	Precision										
	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Average
0.1	0.727	0.660	1.000	0.808	0.821	0.811	0.655	0.735	0.667	1.000	0.788
0.2	0.773	0.679	0.667	0.769	0.692	0.730	0.532	0.533	0.625	1.000	0.700
0.3	0.758	0.566	0.667	0.718	0.615	0.640	0.430	0.410	0.583	0.667	0.605
0.4	0.682	0.618	0.667	0.654	0.571	0.588	0.411	0.371	0.563	0.750	0.587
0.5	0.655	0.581	0.600	0.623	0.518	0.551	0.429	0.347	0.550	0.600	0.545
0.6	0.682	0.547	0.611	0.628	0.538	0.545	0.403	0.323	0.583	0.500	0.536
0.7	0.714	0.523	0.619	0.593	0.527	0.556	0.406	0.309	0.595	0.571	0.542
0.8	0.716	0.531	0.625	0.591	0.529	0.530	0.394	0.319	0.604	0.500	0.534
0.9	0.697	0.501	0.556	0.598	0.513	0.486	0.386	0.314	0.620	0.556	0.523
1.0	0.655	0.475	0.567	0.562	0.508	0.489	0.382	0.313	0.592	0.500	0.504
Average Precision of Google											0.586

Search Engine: MEUFLACC - 6 **Queries: one terms**

Recall	Precision										
	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Average
0.1	0.727	0.698	1.000	0.808	0.821	0.811	0.673	0.723	0.667	1.000	0.793
0.2	0.818	0.736	0.833	0.788	0.705	0.743	0.536	0.538	0.625	1.000	0.732
0.3	0.818	0.579	0.667	0.705	0.667	0.658	0.458	0.426	0.639	0.667	0.628
0.4	0.750	0.665	0.667	0.644	0.577	0.595	0.420	0.364	0.563	0.750	0.599
0.5	0.691	0.604	0.667	0.623	0.528	0.557	0.444	0.354	0.567	0.800	0.583
0.6	0.727	0.544	0.667	0.615	0.526	0.541	0.409	0.321	0.569	0.667	0.559
0.7	0.740	0.509	0.619	0.599	0.524	0.541	0.419	0.306	0.595	0.714	0.557
0.8	0.716	0.526	0.625	0.582	0.513	0.544	0.399	0.305	0.594	0.625	0.543
0.9	0.687	0.499	0.593	0.594	0.516	0.508	0.394	0.295	0.611	0.556	0.525
1.0	0.664	0.472	0.567	0.554	0.500	0.481	0.385	0.293	0.600	0.500	0.502
Average Precision of MEUFLACC – 6											0.602

Note: Q are the queries

Q1= Kasembundit	Q2= Loikrathong	Q3= Nangtalung
Q4= Chartpattana	Q5= Kwanruen	Q6= Baanlaesuan
Q7= Songkarn	Q8= Suvarnabhumi	Q9= Northbangkok
Q10=Nakornphanam		

Search Engine: MEUFLACC - 8 **Queries: one terms**

Recall	Precision										
	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Average
0.1	0.727	0.698	1.000	0.808	0.846	0.811	0.673	0.723	0.667	1.000	0.795
0.2	0.818	0.736	0.833	0.808	0.731	0.757	0.536	0.538	0.625	1.000	0.738
0.3	0.818	0.579	0.778	0.731	0.684	0.658	0.458	0.426	0.667	0.667	0.646
0.4	0.795	0.689	0.750	0.663	0.590	0.601	0.423	0.365	0.646	0.750	0.627
0.5	0.745	0.615	0.667	0.646	0.549	0.568	0.449	0.362	0.600	0.800	0.600
0.6	0.758	0.550	0.722	0.647	0.556	0.563	0.426	0.328	0.597	0.667	0.581
0.7	0.753	0.512	0.667	0.621	0.546	0.560	0.436	0.312	0.595	0.714	0.572
0.8	0.739	0.535	0.667	0.611	0.522	0.564	0.413	0.313	0.604	0.750	0.572
0.9	0.717	0.509	0.630	0.607	0.527	0.526	0.401	0.300	0.620	0.667	0.550
1.0	0.691	0.479	0.600	0.600	0.508	0.492	0.393	0.297	0.625	0.600	0.528
Average Precision of MEUFLACC – 8											0.621

Search Engine: MEUFLACC - 10 **Queries: one terms**

Recall	Precision										
	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Average
0.1	0.727	0.698	1.000	0.808	0.846	0.811	0.673	0.723	0.667	1.000	0.795
0.2	0.818	0.736	0.833	0.808	0.731	0.757	0.536	0.538	0.625	1.000	0.738
0.3	0.818	0.579	0.778	0.731	0.684	0.658	0.458	0.426	0.667	0.667	0.646
0.4	0.818	0.693	0.750	0.663	0.590	0.601	0.423	0.365	0.646	0.750	0.630
0.5	0.764	0.630	0.667	0.646	0.549	0.568	0.449	0.362	0.600	0.800	0.603
0.6	0.758	0.563	0.722	0.647	0.556	0.563	0.426	0.328	0.597	0.667	0.583
0.7	0.753	0.526	0.714	0.621	0.546	0.560	0.436	0.312	0.595	0.714	0.578
0.8	0.761	0.547	0.708	0.611	0.522	0.568	0.413	0.313	0.625	0.750	0.582
0.9	0.737	0.520	0.667	0.624	0.533	0.532	0.401	0.304	0.639	0.667	0.562
1.0	0.709	0.489	0.633	0.612	0.515	0.497	0.393	0.299	0.642	0.600	0.539
Average Precision of MEUFLACC – 10											0.625

Note: Q are the queries

Q1= Kasembundit	Q2= Loikrathong	Q3= Nangtalung
Q4= Chartpattana	Q5= Kwanruen	Q6= Baanlaesuan
Q7= Songkarn	Q8= Suvarnabhumi	Q9= Northbangkok
Q10=Nakornphanam		

Query No. 1: Stang Weight: 0.5
Mongkolsuk Weight: 0.7

Search Engines	Number of relevant documents									
	9	18	27	36	45	54	63	72	81	90
Google	7	14	20	25	29	32	35	37	40	42
Altavista	6	13	18	22	26	29	34	37	40	41
Hotbot	5	13	17	22						
Excite	5	12	16	20	25					
Yahoo	6	12	14							
MEUFLACC – 6	7	14	20	27	32	35	36	38	40	43
MEUFLACC – 8	7	14	20	27	32	35	36	40	43	48
MEUFLACC – 10	7	14	20	27	32	35	36	40	43	48

Query No. 2: Krirk Weight: 0.6
University Weight: 0.4

Search Engines	Number of relevant documents									
	68	136	204	272	340	408	476	544	612	680
Google	50	84	127	146	173	190	213	225	239	250
Altavista	47	74	115	136	157	177	189	212	235	246
Hotbot	45	78	98	107						
Excite	17									
Yahoo	52	80	112	128	151	169	190	219	236	242
MEUFLACC – 6	50	84	127	146	168	174	201	223	238	246
MEUFLACC – 8	50	84	127	146	168	174	201	223	238	246
MEUFLACC – 10	50	84	127	146	168	174	201	223	238	246

Query No. 3: Silpakorn Weight: 0.5
Entrance Weight: 0.8

Search Engines	Number of relevant documents									
	22	44	66	88	110	132	154	176	198	220
Google	16	31	46	52	61	69	75	81	85	89
Altavista	15	28	42	49	56	61	67	75	79	
Hotbot	13	21	35	41	54					
Excite	15	23	33							
Yahoo	13	22	39	47	51	58	64	71	76	82
MEUFLACC – 6	16	31	46	53	59	67	75	81	86	88
MEUFLACC – 8	16	31	46	53	59	67	77	84	88	95
MEUFLACC – 10	16	31	46	53	59	67	77	85	91	97

Query No. 4: Thailand post Weight: 0.4
Stamp Weight: 0.6

Search Engines	Number of relevant documents									
	22	44	66	88	110	132	154	176	198	220
Google	17	30	46	56	71	87	92	98	105	109
Altavista	17	28	42	52	65	77	83	84	93	95
Hotbot	13	18								
Excite	14	18	24							
Yahoo	15	27	39	49	60	69	78			
MEUFLACC – 6	17	30	46	56	71	87	92	98	105	109
MEUFLACC – 8	17	30	46	56	71	87	92	98	107	110
MEUFLACC – 10	17	30	46	56	71	87	92	98	107	110

Query No. 5: Thairakthai Weight: 0.8
Party Weight: 0.4

Search Engines	Number of relevant documents									
	69	138	207	276	345	414	483	552	621	690
Google	42	61	78	86	101	121	144	156	177	186
Altavista	40	53	71	81	97	115	133	143	160	173
Hotbot	38									
Excite	40									
Yahoo	42	55	72	79	91	120	135	147	161	168
MEUFLACC – 6	42	61	78	86	101	124	136	157	174	182
MEUFLACC – 8	44	65	82	89	112	135	147	171	180	189
MEUFLACC – 10	44	65	82	89	112	135	147	171	180	189

Query No. 6: UBCTV Weight: 0.6
Movie Weight: 0.3

Search Engines	Number of relevant documents									
	68	136	204	272	340	408	476	544	612	680
Google	40	54	71	87	96	115	132	141	157	165
Altavista	37	50	68	71	85					
Hotbot	34									
Excite	36									
Yahoo	40	55	67	81	92					
MEUFLACC – 6	40	54	71	87	100	121	134	145	156	168
MEUFLACC – 8	40	54	71	87	100	121	134	145	156	168
MEUFLACC – 10	45	57	74	85	104	124	137	148	154	167

Query No. 7: Kwanruen Weight: 0.7
Magazine Weight: 0.2

Search Engines	Number of relevant documents									
	18	36	54	72	90	108	126	144	162	180
Google	11	17	25	31	37	41	44	48	51	55
Altavista	10	17	26	29	35	41	44	49	52	55
Hotbot	14	19	24	29						
Excite	15									
Yahoo	12	18	25	30	35	39	44	50	52	53
MEUFLACC – 6	11	17	25	31	37	41	47	50	56	61
MEUFLACC – 8	11	17	25	31	37	41	47	50	56	61
MEUFLACC – 10	11	17	25	31	37	41	47	53	58	63

Query No. 8: Suvarnabhumi Weight: 0.5
Airport Weight: 0.3

Search Engines	Number of relevant documents									
	190	380	570	560	950	1140	1330	1520	1710	1900
Google	120	178	245	301	356	398	432	456	476	496
Altavista	112	165	230	278	342	367	411	442	450	470
Hotbot	113	157	175	210	245					
Excite	45									
Yahoo	120	178	242	280	352	371	430	451	462	470
MEUFLACC – 6	125	180	245	295	354	398	442	460	481	501
MEUFLACC – 8	125	180	245	295	354	398	454	470	492	511
MEUFLACC – 10	125	180	245	295	354	398	454	473	495	521

Query No. 9: Bridge Weight: 0.6
Rama8 Weight: 0.7

Search Engines	Number of relevant documents									
	10	20	30	40	50	60	70	80	90	100
Google	6	10	14	18	23	27	34	37	41	44
Altavista	7	13								
Hotbot	6									
Excite	5	11	17							
Yahoo	6	11								
MEUFLACC – 6	6	11	15	18	22	26	34	38	42	44
MEUFLACC – 8	6	11	15	18	22	26	34	39	43	46
MEUFLACC – 10	6	11	15	18	22	26	34	39	43	46

Query No. 10: Thairath Weight: 0.7
Columnist Weight: 0.5

Search Engines	Number of relevant documents									
	3	6	9	12	15	18	21	24	27	30
Google	2	4	5	6	7	8	8	9	11	11
Altavista	2	4	5	5	6					
Hotbot	2	4	6							
Excite	2	4	5	7	9					
Yahoo	2	4	6	6	6					
MEUFLACC – 6	2	4	5	6	7	8	9	9	9	10
MEUFLACC – 8	2	4	5	6	7	8	9	10	10	11
MEUFLACC – 10	2	4	5	6	7	8	9	11	12	12

Search Engine: Hotbot **Queries: two terms**

Recall	Precision										
	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Average
0.1	0.556	0.662	0.591	0.591	0.551	0.500	0.778	0.595	0.600	0.667	0.609
0.2	0.722	0.574	0.477	0.409			0.528	0.413		0.667	0.379
0.3	0.630	0.480	0.530				0.444	0.307		0.667	0.306
0.4	0.611	0.393	0.466				0.403	0.375			0.225
0.5			0.491					0.258			0.075
0.6											
0.7											
0.8											
0.9											
1.0											
Average Precision of Hotbot											0.159

Search Engine: Yahoo **Queries: two terms**

Recall	Precision										
	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Average
0.1	0.667	0.765	0.591	0.682	0.609	0.588	0.667	0.632	0.600	0.667	0.647
0.2	0.667	0.588	0.500	0.614	0.399	0.404	0.500	0.468	0.550	0.667	0.536
0.3	0.519	0.549	0.591	0.591	0.348	0.328	0.463	0.425		0.667	0.448
0.4		0.471	0.534	0.557	0.286	0.298	0.417	0.500		0.500	0.356
0.5		0.444	0.464	0.545	0.264	0.271	0.389	0.371		0.400	0.315
0.6		0.414	0.439	0.523	0.290		0.361	0.325			0.235
0.7		0.399	0.416	0.506	0.280		0.349	0.323			0.227
0.8		0.403	0.403		0.266		0.347	0.297			0.172
0.9		0.386	0.384		0.259		0.321	0.270			0.162
1.0		0.356	0.373		0.243		0.294	0.247			0.151
Average Precision of Yahoo											0.324

Note: Q are the queries

Q1= Stang, Mongkolsuk	Q2= Krirk, University	Q3= Silpakorn, Entrance
Q4= Thailand post, Stamp	Q5= Thairakthai, Party	Q6= UBCTV, Movie
Q7= Kwanruen, Magazine	Q8= Suvarnabhumi, Airport	Q9= Bridge, Rama8
Q10= Thairath, Columnist		

Search Engine: Excite **Queries: two terms**

Recall	Precision										
	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Average
0.1	0.556	0.250	0.682	0.636	0.580	0.529	0.833	0.237	0.500	0.667	0.547
0.2	0.667		0.523	0.409			0.611		0.550	0.667	0.343
0.3	0.593		0.500	0.364			0.500		0.567	0.556	0.308
0.4	0.556						0.458			0.583	0.160
0.5	0.556									0.600	0.116
0.6											
0.7											
0.8											
0.9											
1.0											
Average Precision of Excite											0.147

Search Engine: Altavista **Queries: two terms**

Recall	Precision										
	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Average
0.1	0.667	0.691	0.682	0.773	0.580	0.544	0.556	0.589	0.700	0.667	0.645
0.2	0.722	0.544	0.636	0.636	0.384	0.368	0.472	0.434	0.650	0.667	0.551
0.3	0.667	0.564	0.636	0.636	0.343	0.333	0.481	0.404		0.556	0.462
0.4	0.611	0.500	0.557	0.591	0.293	0.261	0.403	0.496		0.417	0.413
0.5	0.578	0.462	0.509	0.591	0.281	0.250	0.389	0.360		0.400	0.382
0.6	0.537	0.434	0.462	0.583	0.278		0.380	0.322			0.300
0.7	0.540	0.397	0.435	0.539	0.275		0.349	0.309			0.284
0.8	0.514	0.390	0.426	0.477	0.259		0.340	0.291			0.270
0.9	0.494	0.384	0.399	0.470	0.258		0.321	0.263			0.259
1.0	0.456	0.362		0.432	0.251		0.306	0.247			0.205
Average Precision of Altavista											0.377

Note: Q are the queries

Q1= Stang, Mongkolsuk	Q2= Krirk, University	Q3= Silpakorn, Entrance
Q4= Thailand post, Stamp	Q5= Thairakthai, Party	Q6= UBCTV, Movie
Q7= Kwanruen, Magazine	Q8= Suvarnabhumi, Airport	Q9= Bridge, Rama8
Q10= Thairath, Columnist		

Search Engine: Google **Queries: two terms**

Recall	Precision										
	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Average
0.1	0.778	0.735	0.727	0.773	0.609	0.588	0.611	0.632	0.600	0.667	0.672
0.2	0.778	0.618	0.705	0.682	0.442	0.397	0.472	0.468	0.500	0.667	0.573
0.3	0.741	0.623	0.697	0.697	0.377	0.348	0.463	0.430	0.467	0.556	0.540
0.4	0.694	0.537	0.591	0.636	0.312	0.320	0.431	0.538	0.450	0.500	0.501
0.5	0.644	0.509	0.555	0.645	0.293	0.282	0.411	0.375	0.460	0.467	0.464
0.6	0.593	0.466	0.523	0.659	0.292	0.282	0.380	0.349	0.450	0.444	0.444
0.7	0.556	0.447	0.487	0.597	0.298	0.277	0.349	0.325	0.486	0.381	0.420
0.8	0.514	0.414	0.460	0.557	0.283	0.259	0.333	0.300	0.463	0.375	0.396
0.9	0.494	0.391	0.429	0.530	0.285	0.257	0.315	0.278	0.456	0.407	0.384
1.0	0.467	0.368	0.405	0.495	0.270	0.243	0.306	0.261	0.440	0.367	0.362
Average Precision of Google											0.475

Search Engine: MEUFLACC - 6 **Queries: two terms**

Recall	Precision										
	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Average
0.1	0.778	0.735	0.727	0.773	0.609	0.588	0.611	0.658	0.600	0.667	0.675
0.2	0.778	0.618	0.705	0.682	0.442	0.397	0.472	0.474	0.550	0.667	0.578
0.3	0.741	0.623	0.697	0.697	0.377	0.348	0.463	0.430	0.500	0.556	0.543
0.4	0.750	0.537	0.602	0.636	0.312	0.320	0.431	0.527	0.450	0.500	0.506
0.5	0.711	0.494	0.536	0.645	0.293	0.294	0.411	0.373	0.440	0.467	0.466
0.6	0.648	0.426	0.508	0.659	0.300	0.297	0.380	0.349	0.433	0.444	0.444
0.7	0.571	0.422	0.487	0.597	0.282	0.282	0.373	0.332	0.486	0.429	0.426
0.8	0.528	0.410	0.460	0.557	0.284	0.267	0.347	0.303	0.475	0.375	0.401
0.9	0.494	0.389	0.434	0.530	0.280	0.255	0.346	0.281	0.467	0.333	0.381
1.0	0.478	0.362	0.400	0.495	0.264	0.247	0.339	0.264	0.440	0.333	0.362
Average Precision of MEUFLACC – 6											0.478

Note: Q are the queries

Q1= Stang, Mongkolsuk	Q2= Krirk, University	Q3= Silpakorn, Entrance
Q4= Thailand post, Stamp	Q5= Thairakthai, Party	Q6= UBCTV, Movie
Q7= Kwanruen, Magazine	Q8= Suvarnabhumi, Airport	Q9= Bridge, Rama8
Q10= Thairath, Columnist		

Search Engine: MEUFLACC – 8 **Queries: two terms**

Recall	Precision										
	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Average
0.1	0.778	0.735	0.727	0.773	0.638	0.588	0.611	0.658	0.600	0.667	0.677
0.2	0.778	0.618	0.705	0.682	0.471	0.397	0.472	0.474	0.550	0.667	0.581
0.3	0.741	0.623	0.697	0.697	0.396	0.348	0.463	0.430	0.500	0.556	0.545
0.4	0.750	0.537	0.602	0.636	0.322	0.320	0.431	0.527	0.450	0.500	0.508
0.5	0.711	0.494	0.536	0.645	0.325	0.294	0.411	0.373	0.440	0.467	0.470
0.6	0.648	0.426	0.508	0.659	0.326	0.297	0.380	0.349	0.433	0.444	0.447
0.7	0.571	0.422	0.500	0.597	0.304	0.282	0.373	0.341	0.486	0.429	0.431
0.8	0.556	0.410	0.477	0.557	0.310	0.267	0.347	0.309	0.488	0.417	0.414
0.9	0.531	0.389	0.444	0.540	0.290	0.255	0.346	0.288	0.478	0.370	0.393
1.0	0.533	0.362	0.432	0.500	0.274	0.247	0.339	0.269	0.460	0.367	0.378
Average Precision of MEUFLACC – 8											0.484

Search Engine: MEUFLACC - 10 **Queries: two terms**

Recall	Precision										
	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Average
0.1	0.778	0.735	0.727	0.773	0.638	0.662	0.611	0.658	0.600	0.667	0.685
0.2	0.778	0.618	0.705	0.682	0.471	0.419	0.472	0.474	0.550	0.667	0.583
0.3	0.741	0.623	0.697	0.697	0.396	0.363	0.463	0.430	0.500	0.556	0.546
0.4	0.750	0.537	0.602	0.636	0.322	0.313	0.431	0.527	0.450	0.500	0.507
0.5	0.711	0.494	0.536	0.645	0.325	0.306	0.411	0.373	0.440	0.467	0.471
0.6	0.648	0.426	0.508	0.659	0.326	0.304	0.380	0.349	0.433	0.444	0.448
0.7	0.571	0.422	0.500	0.597	0.304	0.288	0.373	0.341	0.486	0.429	0.431
0.8	0.556	0.410	0.483	0.557	0.310	0.272	0.368	0.311	0.488	0.458	0.421
0.9	0.531	0.389	0.460	0.540	0.290	0.252	0.358	0.289	0.478	0.444	0.403
1.0	0.533	0.362	0.441	0.500	0.274	0.246	0.350	0.274	0.460	0.400	0.384
Average Precision of MEUFLACC – 10											0.487

Note: Q are the queries

Q1= Stang, Mongkolsuk	Q2= Krirk, University	Q3= Silpakorn, Entrance
Q4= Thailand post, Stamp	Q5= Thairakthai, Party	Q6= UBCTV, Movie
Q7= Kwanruen, Magazine	Q8= Suvarnabhumi, Airport	Q9= Bridge, Rama8
Q10= Thairath, Columnist		

Query No. 1: Mahidol Weight: 0.5
Library Weight: 0.5
Satang Weight: 0.7

Search Engines	Number of relevant documents									
	4	8	12	16	20	24	28	32	36	40
Google	3	6	8	9	11	12	13	14	14	15
Altavista	3	6	7	8	9	11				
Hotbot	3	4	5	6	6					
Excite	3	4	5	6	6	6	7			
Yahoo	3	5	6	7	7	8				
MEUFLACC – 6	3	6	7	8	9	11	11	12	13	14
MEUFLACC – 8	3	6	7	8	9	11	13	15	19	22
MEUFLACC – 10	3	6	7	8	9	11	13	15	19	22

Query No. 2: Chulalongkorn Weight: 0.2
Entrance Weight: 0.5
Score Weight: 0.7

Search Engines	Number of relevant documents									
	22	44	66	88	110	132	154	176	198	220
Google	14	25	39	52	64	79	85	93	98	101
Altavista	16	24	37	49	58	70	80			
Hotbot	17	25	34							
Excite	17	27	35							
Yahoo	18	26	38	47	53	64	70			
MEUFLACC – 6	14	25	39	52	64	82	90	94	97	101
MEUFLACC – 8	16	27	42	57	71	85	95	103	112	117
MEUFLACC – 10	16	27	42	57	71	85	95	111	119	125

Query No. 3: Suranaree Weight: 0.3
University Weight: 0.4
History Weight: 0.8

Search Engines	Number of relevant documents									
	56	112	168	224	280	336	392	448	504	560
Google	36	57	76	85	96	112	132	147	161	173
Altavista	30	51	67	76	86	102	107			
Hotbot	35	53	63							
Excite	38	47								
Yahoo	32	54	68	80	89	97	106			
MEUFLACC – 6	36	57	76	85	96	112	128	135	153	167
MEUFLACC – 8	36	57	76	85	96	112	128	135	153	167
MEUFLACC – 10	36	57	76	85	96	115	134	142	164	171

Query No. 4: KMUTT Weight: 0.5
Department Weight: 0.7
Electrical Weight: 0.3

Search Engines	Number of relevant documents									
	38	76	114	152	190	228	266	304	342	380
Google	17	30	46	56	71	87	92	98	108	115
Altavista	17	31	44	51	65	75	82	89	96	102
Hotbot	21	28	38							
Excite	21	26								
Yahoo	19	28	37	49	61	74	82	89	96	99
MEUFLACC – 6	17	30	38	45	62	73	83	89	97	103
MEUFLACC – 8	17	30	38	45	62	73	83	89	97	103
MEUFLACC – 10	17	30	38	45	62	73	83	89	97	103

Query No. 5: MCOT Weight: 0.4
Radio Weight: 0.4
Music Weight: 0.9

Search Engines	Number of relevant documents									
	150	300	450	600	750	900	1050	1200	1350	1500
Google	72	102	125	138	150	167	179	186	196	204
Altavista	65	91	116	128	142	158	167	175	183	189
Hotbot	50	67	85	94						
Excite	42									
Yahoo	69	93	119	132	143	157	168	179	191	195
MEUFLACC – 6	72	102	125	138	150	167	179	186	196	204
MEUFLACC – 8	72	102	125	138	150	167	182	192	203	212
MEUFLACC – 10	72	102	125	138	150	167	182	192	203	212

Query No. 6: NECTEC Weight: 0.6
Thaicert Weight: 0.2
Virus Weight: 0.6

Search Engines	Number of relevant documents									
	90	180	270	360	450	540	630	720	810	900
Google	56	87	102	129	143	156	172	180	198	214
Altavista	60	79	98	119	123	136	147	155	163	
Hotbot	45	64	73							
Excite	40									
Yahoo	53	70	76	98	112	127	137	151	157	
MEUFLACC – 6	56	87	102	129	143	156	172	180	198	214
MEUFLACC – 8	56	87	102	129	143	156	172	180	200	216
MEUFLACC – 10	56	87	102	131	146	158	181	189	204	221

Query No. 7: CAT Weight: 0.3
Efone Weight: 0.4
009 Weight: 0.6

Search Engines	Number of relevant documents									
	17	34	51	68	85	102	119	136	153	170
Google	11	17	25	33	38	43	46	48	51	55
Altavista	10	15	22	27	35	41	45	49	51	
Hotbot	10	17								
Excite	10									
Yahoo	10	14	20	27	34	41	45	51	56	62
MEUFLACC – 6	10	16	25	33	36	39	43	45	48	52
MEUFLACC – 8	10	16	25	33	36	39	43	45	48	52
MEUFLACC – 10	10	16	25	33	36	39	43	45	48	52

Query No. 8: Komchadluek Weight: 0.3
News Weight: 0.6
Sports Weight: 0.7

Search Engines	Number of relevant documents									
	68	136	204	272	340	408	476	544	612	680
Google	38	54	69	83	96	115	127	137	149	158
Altavista	38	50	64	76	84	92				
Hotbot	42									
Excite	33									
Yahoo	38	50	60	71	80	89				
MEUFLACC – 6	38	54	69	83	96	115	127	133	145	151
MEUFLACC – 8	43	60	76	94	104	126	134	145	159	164
MEUFLACC – 10	43	60	76	94	104	126	134	145	159	164

Query No. 9: Football Weight: 0.3
Thai Weight: 0.7
Leage Weight: 0.5

Search Engines	Number of relevant documents									
	17	34	51	68	85	102	119	136	153	170
Google	10	16	22	27	32	41	49	54	61	69
Altavista	9	14	20	24	28	36	44	49	52	56
Hotbot	10	16	21	25						
Excite	11	17	24	25						
Yahoo	9	16	22	27	30	36	43	47	51	55
MEUFLACC – 6	10	15	22	27	32	41	49	54	61	69
MEUFLACC – 8	10	15	22	27	32	41	49	54	61	69
MEUFLACC – 10	10	15	22	27	32	44	52	58	64	70

Query No. 10: Dailynews Weight: 0.3
Bangkok Weight: 0.4
Governor Weight: 0.8

Search Engines	Number of relevant documents									
	50	100	150	200	250	300	350	400	450	600
Google	36	62	78	89	103					
Altavista	38	57	68	81	94	110	121	128	135	145
Hotbot	33	49	54							
Excite	38	42								
Yahoo	35	54	68	78	87	94	112	120	133	140
MEUFLACC – 6	36	62	78	89	103	113	121	126	132	135
MEUFLACC – 8	36	62	78	89	112	121	134	141	149	155
MEUFLACC – 10	36	62	78	89	112	121	134	141	149	155

Search Engine: Hotbot **Queries: three terms**

Recall	Precision										
	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Average
0.1	0.75	0.772	0.625	0.552	0.333	0.5	0.588	0.617	0.588	0.66	0.598
0.2	0.5	0.568	0.473	0.368	0.223	0.355	0.5		0.47	0.49	0.394
0.3	0.416	0.515	0.375	0.333	0.188	0.27			0.411	0.36	0.287
0.4	0.375				0.156				0.367		0.08
0.5	0.75	0.772	0.625	0.552	0.333	0.5	0.588	0.617	0.588	0.66	0.598
0.6											
0.7											
0.8											
0.9											
1.0											
Average Precision of Hotbot											0.137

Search Engine: Yahoo **Queries: three terms**

Recall	Precision										
	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Average
0.1	0.75	0.818	0.571	0.5	0.46	0.588	0.588	0.558	0.529	0.7	0.606
0.2	0.625	0.59	0.482	0.368	0.31	0.388	0.411	0.367	0.47	0.54	0.455
0.3	0.5	0.575	0.404	0.324	0.264	0.281	0.392	0.294	0.431	0.453	0.392
0.4	0.437	0.534	0.357	0.322	0.22	0.272	0.397	0.261	0.397	0.39	0.358
0.5	0.35	0.481	0.317	0.321	0.19	0.248	0.4	0.235	0.352	0.348	0.324
0.6	0.333	0.484	0.288	0.324	0.174	0.235	0.401	0.218	0.352	0.313	0.312
0.7		0.454	0.27	0.308	0.16	0.217	0.378		0.361	0.32	0.247
0.8				0.292	0.149	0.209	0.375		0.345	0.3	0.167
0.9				0.28	0.141	0.193	0.366		0.333	0.295	0.161
1.0				0.26	0.13		0.364		0.323	0.233	0.131
Average Precision of Yahoo											0.315

Note: Q are the queries

Q1= Mahidol, Library, Satang
Q2= Chulalongkorn, Entrance, Score
Q3= Suranaree, University, History
Q4= KMUTT, Department, Electrical
Q5= MCOT, Radio, Music
Q6= NECTEC, Thaicert, Virus
Q7= CAT, Efone, 009
Q8= Komchadluek, News, Sports
Q9= Football, Thai, Leage
Q10=Dailynews, Bangkok, Governor

Search Engine: Excite **Queries: three terms**

Recall	Precision										
	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Average
0.1	0.75	0.772	0.678	0.552	0.28	0.444	0.588	0.485	0.647	0.76	0.595
0.2	0.5	0.613	0.419	0.342					0.5	0.42	0.279
0.3	0.416	0.53							0.47		0.141
0.4	0.375								0.367		0.07
0.5	0.3										
0.6	0.25										0.109
0.7	0.25										
0.8											
0.9											
1.0											
Average Precision of Excite											0.109

Search Engine: Altavista **Queries: three terms**

Recall	Precision										
	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Average
0.1	0.75	0.727	0.535	0.477	0.433	0.666	0.588	0.558	0.529	0.76	0.599
0.2	0.75	0.545	0.455	0.407	0.303	0.438	0.441	0.367	0.411	0.57	0.469
0.3	0.583	0.56	0.398	0.385	0.257	0.362	0.431	0.313	0.392	0.453	0.414
0.4	0.5	0.556	0.339	0.335	0.213	0.33	0.397	0.279	0.352	0.405	0.37
0.5	0.45	0.527	0.307	0.342	0.189	0.273	0.411	0.247	0.329	0.376	0.345
0.6	0.458	0.53	0.303	0.328	0.175	0.251	0.401	0.225	0.352	0.366	0.339
0.7		0.519	0.272	0.308	0.159	0.233	0.378		0.369	0.345	0.258
0.8				0.292	0.145	0.215	0.36		0.36	0.32	0.169
0.9				0.28	0.135	0.201	0.333		0.339	0.3	0.159
1.0				0.268	0.126				0.329	0.241	0.09
Average Precision of Altavista											0.322

Note: Q are the queries

Q1= Mahidol, Library, Satang	Q2= Chulalongkorn, Entrance, Score
Q3= Suranaree, University, History	Q4= KMUTT, Department, Electrical
Q5= MCOT, Radio, Music	Q6= NECTEC, Thaicert, Virus
Q7= CAT, Efone, 009	Q8= Komchadluek, News, Sports
Q9= Football, Thai, Leage	Q10=Dailynews, Bangkok, Governor

Search Engine: Google **Queries: three terms**

Recall	Precision										
	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Average
0.1	0.75	0.636	0.642	0.447	0.48	0.622	0.647	0.558	0.588	0.72	0.609
0.2	0.75	0.568	0.508	0.394	0.34	0.483	0.5	0.397	0.47	0.62	0.503
0.3	0.666	0.59	0.452	0.403	0.277	0.377	0.49	0.338	0.431	0.52	0.454
0.4	0.562	0.59	0.379	0.368	0.23	0.358	0.485	0.305	0.397	0.445	0.412
0.5	0.55	0.581	0.342	0.373	0.2	0.317	0.447	0.282	0.376	0.412	0.388
0.6	0.5	0.598	0.333	0.381	0.185	0.288	0.421	0.281	0.401		0.339
0.7	0.464	0.551	0.336	0.345	0.17	0.273	0.386	0.266	0.411		0.32
0.8	0.437	0.528	0.328	0.322	0.155	0.25	0.352	0.251	0.397		0.302
0.9	0.388	0.494	0.319	0.315	0.145	0.244	0.333	0.243	0.398		0.288
1.0	0.375	0.459	0.308	0.302	0.136	0.237	0.323	0.232	0.405		0.278
Average Precision of Google											0.389

Search Engine: MEUFLACC - 6 **Queries: three terms**

Recall	Precision										
	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Average
0.1	0.75	0.636	0.642	0.447	0.48	0.622	0.588	0.558	0.588	0.72	0.603
0.2	0.75	0.568	0.508	0.394	0.34	0.483	0.47	0.397	0.441	0.62	0.497
0.3	0.583	0.59	0.452	0.333	0.277	0.377	0.49	0.338	0.431	0.52	0.439
0.4	0.5	0.59	0.379	0.296	0.23	0.358	0.485	0.305	0.397	0.445	0.398
0.5	0.45	0.581	0.342	0.326	0.2	0.317	0.423	0.282	0.376	0.412	0.371
0.6	0.458	0.621	0.333	0.32	0.185	0.288	0.382	0.281	0.401	0.376	0.365
0.7	0.392	0.584	0.326	0.312	0.17	0.273	0.361	0.266	0.411	0.345	0.344
0.8	0.375	0.534	0.301	0.292	0.155	0.25	0.33	0.244	0.397	0.315	0.319
0.9	0.361	0.489	0.303	0.283	0.145	0.244	0.313	0.236	0.398	0.293	0.307
1.0	0.35	0.459	0.298	0.271	0.136	0.237	0.305	0.222	0.405	0.225	0.291
Average Precision of MEUFLACC – 6											0.393

Note: Q are the queries

Q1= Mahidol, Library, Satang	Q2= Chulalongkorn, Entrance, Score
Q3= Suranaree, University, History	Q4= KMUTT, Department, Electrical
Q5= MCOT, Radio, Music	Q6= NECTEC, Thaicert, Virus
Q7= CAT, Efone, 009	Q8= Komchadluek, News, Sports
Q9= Football, Thai, Leage	Q10=Dailynews, Bangkok, Governor

Search Engine: MEUFLACC – 8 **Queries: three terms**

Recall	Precision										
	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Average
0.1	0.75	0.727	0.642	0.447	0.48	0.622	0.588	0.632	0.588	0.72	0.619
0.2	0.75	0.613	0.508	0.394	0.34	0.483	0.47	0.441	0.441	0.62	0.506
0.3	0.583	0.636	0.452	0.333	0.277	0.377	0.49	0.372	0.431	0.52	0.447
0.4	0.5	0.647	0.379	0.296	0.23	0.358	0.485	0.345	0.397	0.445	0.408
0.5	0.45	0.645	0.342	0.326	0.2	0.317	0.423	0.305	0.376	0.448	0.383
0.6	0.458	0.643	0.333	0.32	0.185	0.288	0.382	0.308	0.401	0.403	0.372
0.7	0.464	0.616	0.326	0.312	0.173	0.273	0.361	0.281	0.411	0.382	0.36
0.8	0.468	0.585	0.301	0.292	0.16	0.25	0.33	0.266	0.397	0.352	0.34
0.9	0.527	0.565	0.303	0.283	0.15	0.246	0.313	0.259	0.398	0.331	0.338
1.0	0.55	0.531	0.298	0.271	0.141	0.24	0.305	0.241	0.405	0.258	0.324
Average Precision of MEUFLACC – 8											0.41

Search Engine: MEUFLACC - 10 **Queries: three terms**

Recall	Precision										
	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Average
0.1	0.75	0.727	0.642	0.447	0.48	0.622	0.588	0.632	0.588	0.72	0.619
0.2	0.75	0.613	0.508	0.394	0.34	0.483	0.47	0.441	0.441	0.62	0.506
0.3	0.583	0.636	0.452	0.333	0.277	0.377	0.49	0.372	0.431	0.52	0.447
0.4	0.5	0.647	0.379	0.296	0.23	0.363	0.485	0.345	0.397	0.445	0.409
0.5	0.45	0.645	0.342	0.326	0.2	0.324	0.423	0.305	0.376	0.448	0.384
0.6	0.458	0.643	0.342	0.32	0.185	0.292	0.382	0.308	0.431	0.403	0.376
0.7	0.464	0.616	0.341	0.312	0.173	0.287	0.361	0.281	0.436	0.382	0.365
0.8	0.468	0.63	0.316	0.292	0.16	0.262	0.33	0.266	0.426	0.352	0.35
0.9	0.527	0.601	0.325	0.283	0.15	0.251	0.313	0.259	0.418	0.331	0.346
1.0	0.55	0.568	0.305	0.271	0.141	0.245	0.305	0.241	0.411	0.258	0.329
Average Precision of MEUFLACC – 10											0.413

Note: Q are the queries

Q1= Mahidol, Library, Satang	Q2= Chulalongkorn, Entrance, Score
Q3= Suranaree, University, History	Q4= KMUTT, Department, Electrical
Q5= MCOT, Radio, Music	Q6= NECTEC, Thaicert, Virus
Q7= CAT, Efone, 009	Q8= Komchadluek, News, Sports
Q9= Football, Thai, Leage	Q10=Dailynews, Bangkok, Governor

Search Engine: ODP

	Cluster Group	**Recall**					
		0.1	**0.2**	**0.3**	**0.4**	**0.5**	
Query: Mobile computing All clusters:16 Useful clusters:7	Wearable	1	0.666	0.75	0.571	0.5	Precision
	Wireless data	1	0.5	0.6	0.666	0.625	
	Notebook& Laptop	0.333	0.4	0.5	0.571	0.625	
	Palm& Handheld	1	1	0.75	0.8	0.714	
Query: Operating System All clusters:107 Useful clusters:32	Mainframe	0.25	0.4	0.48	0.444	0.416	
	Netware	0.5	0.5	0.5	0.4	0.333	
	FreeBSD	0.333	0.5	0.6	0.571	0.333	
	Windows	0.5	0.333	0.48	0.4	0.384	
Query: Security All clusters:23 Useful clusters:10	IDS	0.5	0.333	0.428	0.444	0.454	
	Firewall	0.5	0.5	0.375	0.4	0.357	
	Antivirus	1	0.4	0.5	0.5	0.555	
	Biometrics	0.333	0.285	0.375	0.363	0.384	
Average Precision		0.604	0.484	0.528	0.51	0.473	
F-measure		0.171	0.283	0.382	0.448	0.486	

Search Engine: Yahoo

Query	**Cluster Group**	**Recall 0.1**	**0.2**	**0.3**	**0.4**	**0.5**	
Query: Mobile computing All clusters:10 Useful clusters:4	Wearable	0.5	0.4	0.5	0.5	0.555	Precision
	Wireless	1	0.666	0.375	0.4	0.416	
	Tablet computer	0.333	0.4	0.5	0.5	0.454	
	PDA	0.25	0.4	0.375	0.4	0.333	
Query: Operating System All clusters:40 Useful clusters:8	Linux	0.333	0.25	0.3	0.307	0.294	
	UNIX	0.2	0.333	0.375	0.4	0.294	
	Operating system	0.333	0.285	0.375	0.333	0.333	
	Windows	0.2	0.25	0.333	0.4	0.25	
Query: Security All clusters:37 Useful clusters:8	Hacking	0.25	0.4	0.375	0.363	0.357	
	Firewall	0.2	0.285	0.3	0.333	0.333	
	Virus	0.333	0.222	0.23	0.285	0.357	
	IDS	0.25	0.285	0.333	0.307	0.294	
Average Precision		0.348	0.348	0.364	0.377	0.355	
F-measure		0.155	0.254	0.328	0.388	0.415	

Search Engine: MEUFLACC-6

	Cluster Group	**Recall**					
		0.1	**0.2**	**0.3**	**0.4**	**0.5**	
Query: Mobile computing All clusters: Useful clusters:	Wearable	1	1	0.75	0.571	0.5	Precision
	Wireless	1	0.5	0.6	0.571	0.625	
	Tablet computer	1	0.5	0.75	0.666	0.5	
	PDA	1	1	0.5	0.5	0.454	
Query: Operating System All clusters: Useful clusters:	Linux	0.25	0.4	0.428	0.444	0.416	
	UNIX	0.333	0.4	0.5	0.4	0.333	
	Operating system	0.5	0.666	0.6	0.571	0.555	
	Windows	0.5	0.333	0.428	0.444	0.384	
Query: Security All clusters: Useful clusters:	Hacking	0.333	0.4	0.375	0.444	0.357	
	Firewall	0.5	0.666	0.6	0.444	0.454	
	Virus	1	0.5	0.5	0.571	0.416	
	IDS	0.333	0.4	0.5	0.5	0.555	
Average Precision		0.645	0.563	0.544	0.51	0.462	
F-measure		0.173	0.295	0.386	0.448	0.48	

Search Engine: MEUFLACC-8

Query	**Cluster Group**	**Recall**					
		0.1	**0.2**	**0.3**	**0.4**	**0.5**	
Query: Mobile computing All clusters: Useful clusters:	Wearable	1	1	0.75	0.8	0.833	Precision
	Wireless	1	0.5	0.6	0.571	0.625	
	Tablet computer	1	0.5	0.6	0.666	0.555	
	PDA	1	1	0.6	0.666	0.714	
Query: Operating System All clusters: Useful clusters:	Linux	0.5	0.666	0.428	0.444	0.5	
	UNIX	0.333	0.4	0.428	0.5	0.555	
	Operating system	1	0.666	0.6	0.666	0.555	
	Windows	1	0.5	0.6	0.444	0.5	
Query: Security All clusters: Useful clusters:	Hacking	0.333	0.5	0.428	0.5	0.5	
	Firewall	0.5	0.666	0.6	0.444	0.454	
	Virus	1	1	0.75	0.571	0.416	
	IDS	0.333	0.4	0.5	0.5	0.555	
Average Precision		0.749	0.649	0.573	0.564	0.563	
F-measure		0.176	0.305	0.393	0.468	0.529	

Search Engine: MEUFLACC-10

Query	**Cluster Group**	**Recall**					
		0.1	**0.2**	**0.3**	**0.4**	**0.5**	
Query: Mobile computing All clusters: Useful clusters:	Wearable Computing	1	1	0.75	0.8	0.833	Precision
	Wireless Networking	1	0.5	0.6	0.666	0.714	
	Tablet computer	1	0.5	1	0.666	0.555	
	PDA	1	1	0.6	0.666	0.714	
Query: Operating System All clusters: Useful clusters:	BeOS	0.5	0.666	0.75	0.444	0.5	
	UNIX	1	0.666	0.6	0.571	0.555	
	X-Windows	1	1	0.6	0.666	0.555	
	Windows	1	0.5	0.6	0.571	0.625	
Query: Security All clusters: Useful clusters:	Hacking	0.333	0.5	0.428	0.5	0.5	
	Firewall	0.5	0.666	0.6	0.666	0.555	
	Virus &Worms	1	1	0.75	0.8	0.833	
	Cryptography	0.5	0.4	0.5	0.5	0.555	
Average Precision		0.819	0.699	0.648	0.626	0.624	
F-measure		0.178	0.311	0.41	0.488	0.555	

Query No. 1: XML

Search Engines	Order of documents									
	1	2	3	4	5	6	7	8	9	10
CBCMSE-3	6	8	17	21	22	26	28	32	35	42
CBCMSE-5	1	2	4	5	13	25	26	30	32	41
CBCMSE-7	1	2	9	11	20	24	25	33	36	40
MEUFLACC – 3	1	2	4	5	6	9	11	12	13	15
MEUFLACC – 5	1	2	4	5	6	7	8	10	14	16
MEUFLACC – 7	1	2	4	5	6	8	9	10	12	15

Keyphrase Extractions	Number of	Number of viewed clusters				
		1	2	3	4	5
CBCMSE-3	Relevant documents	9	1			
	All documents	39	4			
CBCMSE-5	Relevant documents	4	6			
	All documents	6	40			
CBCMSE-7	Relevant documents	2	8			
	All documents	3	40			
MEUFLACC – 3	Relevant documents	8	2			
	All documents	18	5			
MEUFLACC – 5	Relevant documents	8	3			
	All documents	20	6			
MEUFLACC – 7	Relevant documents	8	3			
	All documents	20	6			

Query No. 2: Text clustering algorithm

Search Engines	Order of documents									
	1	2	3	4	5	6	7	8	9	10
CBCMSE-3	3	4	7	12	15	20	22	24	31	32
CBCMSE-5	3	4	5	11	12	13	15	21	23	28
CBCMSE-7	1	2	6	10	11	12	13	20	22	28
MEUFLACC – 3	1	2	4	5	6	8	10	12	13	14
MEUFLACC – 5	2	3	4	7	8	9	10	12	14	15
MEUFLACC – 7	1	2	4	5	6	8	10	12	13	14

Keyphrase Extractions	Number of	Number of viewed clusters				
		1	2	3	4	5
CBCMSE-3	Relevant documents	4	1	4		
	All documents	19	2	10		
CBCMSE-5	Relevant documents	2	1	7		
	All documents	4	2	24		
CBCMSE-7	Relevant documents	2	1	7		
	All documents	4	2	24		
MEUFLACC – 3	Relevant documents	4	2	2	3	
	All documents	10	5	4	4	
MEUFLACC – 5	Relevant documents	4	1	1	5	
	All documents	15	5	2	14	
MEUFLACC – 7	Relevant documents	6	5			
	All documents	13	7			

Query No. 3: WAP

Search Engines	Order of documents									
	1	2	3	4	5	6	7	8	9	10
CBCMSE-3	1	2	3	4	6	7	10	16	19	24
CBCMSE-5	1	2	3	4	6	7	12	14	20	27
CBCMSE-7	1	3	5	7	8	9	10	11	13	23
MEUFLACC – 3	1	2	3	4	9	11	14	15	17	18
MEUFLACC – 5	1	2	3	4	8	9	12	15	17	18
MEUFLACC – 7	1	2	3	4	5	7	12	14	17	18

Keyphrase Extractions	Number of	Number of viewed clusters				
		1	2	3	4	5
CBCMSE-3	Relevant documents	4	2	1	3	
	All documents	4	4	3	20	
CBCMSE-5	Relevant documents	4	2	1	1	2
	All documents	4	7	2	2	20
CBCMSE-7	Relevant documents	1	1	6	1	1
	All documents	2	2	7	7	16
MEUFLACC – 3	Relevant documents	14				
	All documents	25				
MEUFLACC – 5	Relevant documents	16				
	All documents	29				
MEUFLACC – 7	Relevant documents	18				
	All documents	32				

Query No. 4: ERP

Search Engines	Order of documents									
	1	2	3	4	5	6	7	8	9	10
CBCMSE-3	2	4	9	11						
CBCMSE-5	1	2	4	10						
CBCMSE-7	2	5	6	8						
MEUFLACC – 3	2	4	5	6	7	8	11	14	16	18
MEUFLACC – 5	2	4	5	6	7	8	11	14	15	18
MEUFLACC – 7	2	4	5	6	7	9	10	13	15	16

Keyphrase Extractions	Number of	Number of viewed clusters				
		1	2	3	4	5
CBCMSE-3	Relevant documents	4				
	All documents	11				
CBCMSE-5	Relevant documents	2	3			
	All documents	2	11			
CBCMSE-7	Relevant documents	3	3			
	All documents	6	11			
MEUFLACC – 3	Relevant documents	4	6			
	All documents	15	19			
MEUFLACC – 5	Relevant documents	7	3			
	All documents	17	10			
MEUFLACC – 7	Relevant documents	8	6			
	All documents	20	17			

Query No. 5: CORBA

Search Engines	Order of documents									
	1	2	3	4	5	6	7	8	9	10
CBCMSE-3	1	2	3	4	5	7	8	9	11	12
CBCMSE-5	1	2	3	4	5	7	8	9	11	12
CBCMSE-7	1	2	3	4	5	7	8	9	11	12
MEUFLACC – 3	1	2	4	7	8	9	12	14	16	18
MEUFLACC – 5	1	2	3	5	6	9	12	13	14	16
MEUFLACC – 7	1	2	4	5	6	7	9	10	14	15

Keyphrase Extractions	Number of	Number of viewed clusters				
		1	2	3	4	5
CBCMSE-3	Relevant documents	13				
	All documents	21				
CBCMSE-5	Relevant documents	13				
	All documents	21				
CBCMSE-7	Relevant documents	13				
	All documents	21				
MEUFLACC – 3	Relevant documents	3	4	4		
	All documents	6	12	10		
MEUFLACC – 5	Relevant documents	7	4			
	All documents	14	16			
MEUFLACC – 7	Relevant documents	8	5			
	All documents	18	18			

Query No. 6: UML

Search Engines	Order of documents									
	1	2	3	4	5	6	7	8	9	10
CBCMSE-3	2	3	4	5	7	8	10	11	23	24
CBCMSE-5	2	3	4	5	7	8	10	11	24	25
CBCMSE-7	2	3	4	5	7	8	10	22	24	25
MEUFLACC – 3	2	4	5	8	9	10	13	15	16	19
MEUFLACC – 5	1	2	3	5	7	10	11	12	15	16
MEUFLACC – 7	1	2	3	4	7	10	11	12	14	16

Keyphrase Extractions	Number of	Number of viewed clusters				
		1	2	3	4	5
CBCMSE-3	Relevant documents	8	2			
	All documents	20	9			
CBCMSE-5	Relevant documents	8	2			
	All documents	21	5			
CBCMSE-7	Relevant documents	8	2			
	All documents	21	5			
MEUFLACC – 3	Relevant documents	12				
	All documents	20				
MEUFLACC – 5	Relevant documents	15				
	All documents	28				
MEUFLACC – 7	Relevant documents	17				
	All documents	30				

Query No. 7: CDMA

Search Engines	Order of documents									
	1	2	3	4	5	6	7	8	9	10
CBCMSE-3	2	3	4	6	7	8	12	15	16	18
CBCMSE-5	2	3	4	6	7	8	9	15	16	18
CBCMSE-7	2	3	4	6	7	8	9	15	16	18
MEUFLACC – 3	1	2	3	5	9	12	13	14	18	19
MEUFLACC – 5	1	2	3	5	9	12	13	14	18	19
MEUFLACC – 7	1	2	3	5	6	7	8	10	12	14

Keyphrase Extractions	Number of	Number of viewed clusters				
		1	2	3	4	5
CBCMSE-3	Relevant documents	7	1	3		
	All documents	11	1	13		
CBCMSE-5	Relevant documents	8	3			
	All documents	12	14			
CBCMSE-7	Relevant documents	8	3			
	All documents	12	11			
MEUFLACC – 3	Relevant documents	10				
	All documents	20				
MEUFLACC – 5	Relevant documents	12				
	All documents	16				
MEUFLACC – 7	Relevant documents	12				
	All documents	18				

Query No. 8: FDDI

Search Engines	Order of documents									
	1	2	3	4	5	6	7	8	9	10
CBCMSE-3	1	2	3	4	5	6	7	11	13	14
CBCMSE-5	1	2	3	4	5	6	12	14	15	20
CBCMSE-7	1	2	3	4	5	6	7	14	16	17
MEUFLACC – 3	1	2	3	4	6	7	8	11	15	16
MEUFLACC – 5	1	2	3	4	6	7	8	10	13	14
MEUFLACC – 7	1	2	3	4	7	8	9	11	13	14

Keyphrase Extractions	Number of	Number of viewed clusters				
		1	2	3	4	5
CBCMSE-3	Relevant documents	10				
	All documents	18				
CBCMSE-5	Relevant documents	9	1			
	All documents	19	3			
CBCMSE-7	Relevant documents	10				
	All documents	21				
MEUFLACC – 3	Relevant documents	10				
	All documents	16				
MEUFLACC – 5	Relevant documents	13				
	All documents	19				
MEUFLACC – 7	Relevant documents	14				
	All documents	22				

Query No. 9: RMON

Search Engines	Order of documents									
	1	2	3	4	5	6	7	8	9	10
CBCMSE-3	1	2	4	5	6	7	8	9	11	12
CBCMSE-5	1	2	4	5	6	7	8	9	11	12
CBCMSE-7	1	2	4	5	6	7	8	9	11	12
MEUFLACC – 3	1	3	4	7	8	9	10	12	13	15
MEUFLACC – 5	1	3	4	7	8	9	10	12	13	15
MEUFLACC – 7	1	3	4	7	8	9	10	12	13	15

Keyphrase Extractions	Number of	Number of viewed clusters				
		1	2	3	4	5
CBCMSE-3	Relevant documents	10				
	All documents	23				
CBCMSE-5	Relevant documents	10				
	All documents	23				
CBCMSE-7	Relevant documents	10				
	All documents	23				
MEUFLACC – 3	Relevant documents	6				
	All documents	10				
MEUFLACC – 5	Relevant documents	11				
	All documents	14				
MEUFLACC – 7	Relevant documents	14				
	All documents	16				

Query No. 10: SLIP

Search Engines	Order of documents									
	1	2	3	4	5	6	7	8	9	10
CBCMSE-3	1	2	3	5	6	8	10	15	16	18
CBCMSE-5	1	2	3	5	7	9	11	16	17	18
CBCMSE-7	1	2	3	5	6	8	10	12	18	19
MEUFLACC – 3	1	2	3	6	7	9	10	11	14	17
MEUFLACC – 5	1	2	5	6	7	9	10	11	14	17
MEUFLACC – 7	1	2	4	5	6	7	9	10	12	13

Keyphrase Extractions	Number of	Number of viewed clusters				
		1	2	3	4	5
CBCMSE-3	Relevant documents	7	3			
	All documents	14	7			
CBCMSE-5	Relevant documents	7	3			
	All documents	14	6			
CBCMSE-7	Relevant documents	8	5			
	All documents	17	6			
MEUFLACC – 3	Relevant documents	12				
	All documents	16				
MEUFLACC – 5	Relevant documents	15				
	All documents	20				
MEUFLACC – 7	Relevant documents	15				
	All documents	20				

BIOGRAPHY

NAME	Mr. Werasak Yuokoolbodee
DATE OF BIRTH	9 March 1974
PLACE OF BIRTH	Chumphon, Thailand
INSTITUTIONS ATTENDED	Rajamangala Institute of Technology,1994-1996:
	Bachelor of Engineer (Electrical Engineering)
	Mahidol University, 2000-2003:
	Master of Science (Computer Science)
POSITION & OFFICE	1998-2001,
	Rajamangala Institute of Technology
	Bangkok Campus.
	Bangkok, Thailand.
	Position : Lecturer.
	2001-Present,
	Provincial Waterwork Authority.
	Bangkok, Thailand.
	Position : Engineer.