Observations on Design and Architecture of Meta-Search Engine

Dissertation submitted to the Jawaharlal Nehru University

in partial fulfillment of the requirements

for the award of Degree of

MASTER OF TECHNOLOGY
IN
Computer Science and Technology

By

Vajenti Mala

Under the guidance of

Dr. Aditi Sharan

SCHOOL OF COMPUTER AND SYSTEMS SCIENCES

JAWAHARLAL NEHRU UNIVERSITY

NEW DELHI, 110067 (INDIA)

July, 2010

ज. ने. वि.

JNU

SCHOOL OF COMPUTER AND SYSTEMS SCIENCES

JAWAHARLAL NEHRU UNIVERSITY

NEW DELHI, 110067 (INDIA)

DECLARATION

I hereby declare that this dissertation entitled **"Observations on Design and Architecture of Meta-Search Engine"** submitted by me to the School of Computer and systems Sciences, Jawaharlal Nehru University, New Delhi for the award of **MASTER OF TECHNOLOGY IN COMPUTER SCIENCE AND TECHNOLOGY** is a bonafide work carried out by me under the supervision of Dr Aditi Sharan.

The matter embodied in this dissertation has not been submitted to any other University or Institution for the award of any other degree or diploma.

Vajenti Mala

SCHOOL OF COMPUTER AND SYSTEMS SCIENCES

JAWAHARLAL NEHRU UNIVERSITY

NEW DELHI, 110067 (INDIA)

ज. ने. वि.
JNU

SCHOOL OF COMPUTER AND SYSTEMS SCIENCES

JAWAHARLAL NEHRU UNIVERSITY

NEW DELHI, 110067 (INDIA)

CERTIFICATE

This is certify that this dissertation titled **"Observations on Design and Architecture of Meta-Search Engine"** submitted by Vajenti Mala to the School of Computer and systems Sciences, Jawaharlal Nehru University, New Delhi for the award of **MASTER OF TECHNOLOGY IN COMPUTER SCIENCE AND TECHNOLOGY** is a bonafide work carried out by her under my supervision.

The matter embodied in this dissertation has not been submitted to any other University or Institution for the award of any other degree or diploma.

Dean

(SC&SS)

Jawaharlal Nehru University

New Delhi-110067(India)

Sharan

Dr. Aditi Sharan (Supervisor)

Assistant Professor, SC&SS

Jawaharlal Nehru University

New Delhi-110067(India)

Dedicated to

My parents and all who love me the most…

Acknowledgements

At the very outset I would like to thank Almighty God for all the favors He showered upon me throughout my life.

I owe my heartiest gratitude to my supervisor, **Dr Aditi Sharan.** Her empowering supervision, timely guidance and continuous support facilitated my progress and made me to complete the dissertation successfully. Had it not been for her persistent motivation and inspiration, it was impossible for me to bring out the dissertation in its present form. She inculcated in me the desire to learn and explore. She was always reachable for me.

Without mentioning names, I would, in particular, like to thank all the faculty members of **School of Computer and Systems Sciences** for their immense cooperation, help and encouragement.

I also want to remember and thank all the non-teaching and administrative staff of the **School of Computer & SystemsSciences** for their generous behavior and cooperation throughout my course of M.Tech.

Thanks are due to administration of JNU for providing a congenial environment for making our work a success. Their academic support has been a real asset in completing this work.

Thanks are due to all my classmates and friends who stood by me through thick and thin.

I am deeply indebted to my parents, my brothers (Vijay Anand and Kailash Anand) and sisters (Devi, Kamla devi, Shanti, Gomanti, Poonam, and Mohani) for their love, care, support and encouragement throughout of my life. None of this would be possible without their help and assistance and "thanks" is too little a word to express my feeling. Last but not the least, my thanks also go to all those who were of help to me during my course. Words fail me to thank them all.

Over the years, they cheer for even a tiny progress I made and always have faith in me no matter how difficult life is.

Vajenti Mala

ABSTRACT

In recent years World Wide Web has become largest and most widely used repository to obtain information in various fields. However it is being said about web that a person exploring the web is drowning in the information but staring for the knowledge. It is becoming increasing difficult to extract relevant information from the web, specifically if the information relates to a specific topic. The information on the web is organized in the form of hyperlinked web pages and a web search engine is essentially an information retrieval system for web pages. However, Web pages have several features that are not usually associated with documents in traditional IR systems and these features have been explored by search engine developers to improve the retrieval effectiveness of search engines. Thus search engine has some limitations. This has led to development of meta-search engine.

A user's information needs are stored in the databases of multiple search engines. To support the unified access to multiple search engines, a meta-search engine can be constructed. When meta-search engine receives a query from a user, it invokes the underlying search engine to retrieve useful information for the user. Meta-search engine is a search tool which increases the search coverage of the web and improving the scalability of the search engine.

In this dissertation I have focused on architecture of meta-search engine and its working. Several underlying challenges for building a good meta-search engine have been discussed. Among the main challenges, the database selection problem is to identify the relevant information according to the user needs. The document selection problem is to determine what documents to retrieve from each identify databases and result merging problem is to combine the documents returned from multiple search engines. Some experiments have been performed on document selection approaches. An important observation is that for most of the queries overlapping in the result sets are very less, therefore there is a large scope of selecting appropriate approaches to fulfill user's need in more efficient way. This is an emerging area of research area, which has a lot of open problems; my work tries to bring out some of these issues.

Table of Contents

Page no.

List of Figures

List of Tables

CHAPTER 1 Introduction to Meta Search Engine

1.1. Introduction

The web has become a vast information resource as millions of people are using the internet on a regular basis. Various types of IR (Information Retrieval) tools may help to get the information on internet. Web search engine, Meta search engine and directories are three major types of web services for retrieving information from Web. Our work is concerned with study of design and architecture of meta-search engines. Meta search engine is a system that supports unified access to multiple search engines or databases. This chapter introduces meta search engines and discusses their applications. But before introducing meta search engine, we discuss about search engines, their role in retrieving information from Web as well as their limitations. Limitations of search engine allow us to focus on the need of meta search engines.

1.2 Search Engine:

A web search engine is essentially an information retrieval system for web pages. [5] Search engines are used for full-text searchers of web pages. Search engines are best at finding unique keywords, phrases, quotes and information buried in the full-text of web pages. These engines maintain large database of web pages files, which is crawled and indexed off line. When user submits a query, the index of database is searched and based on the match between query and document index, ranked list of relevant documents is returned to user. Search engines provide is a good way to find a wide range of responses to specific queries.

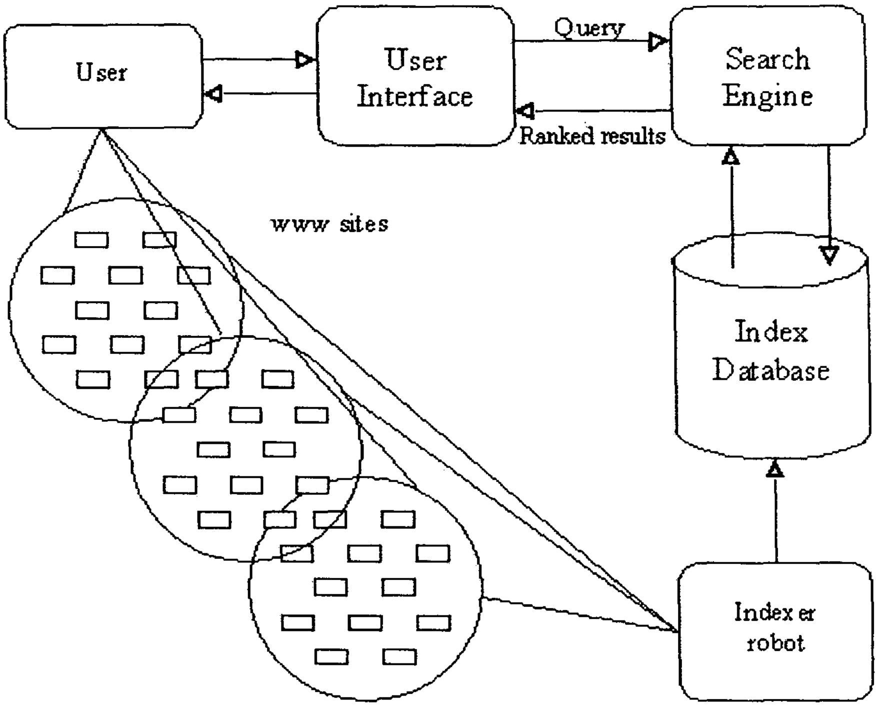

Fig: 1.1 Standard Architecture of Search Engine

A search engine consists of two parts, viz. a back end database (server side) and a GUI (client side), to facilitate the user to type the search term. On the server side, the process involves creation of a database and its periodic updating done by software called Spider. The spider "crawls" the URL pages periodically and indexes the crawled pages in the database. Most search engines use a centralized crawler indexer architecture .The interface between the client and server side consists of matching the posted query with the entries in the database and retrieving the matched URLs to the user's machine. Some of the examples of popular Search engines are : www.yahoo.com, www.google.com

www.msn.com,www.altavista.com

1.2.1 Role of Search Engines for Web search

Search engines play a very important role for searching information from the web. Efficiency of any web based IR system heavily depends on the efficiency of underlying search engine which is being used by that system. Some important functions of search engines are as follows:

- To display specific relevant pages from the billons of web pages available on the web in which users are interested.
- To support relatively complex queries and break them into formal query.
- Providing relatively quick response to produce meaningful information from World Wide Web.
- Quick and meaningful response of complex queries.

Broadly speaking, there are two categories of search engines.

1. General purpose search engines: - The objective of general purpose search is to provide as many related web pages web pages as possible based on user's query. AltaVista, Excite, Lycos, Google, World Wide Web Worm (wwww) etc... are some example of general purpose search engines.
2. Special purpose search engines: - The special-purpose search engines, on the other hand, focus on searching web pages on particular topics. Instead of searching horizontally to increase coverage of results these engines perform more focused link based search, which is also called vertical search. For example, the Medical World Search (www.mwsearch.com) is a search engine for medical information and Movie Review Query Engine (www.mrqe.com) let the users to search for movie reviews.

   Along with the usability of both types of search engines, search engines have some challenges and have their own limitations.

1.2.2 Challenges for Search engines

- ***Understanding*** of user's need (based on user's query) and returning the relevant web pages is a major challenge for any search engine.
- To rank the web documents according to their actual importance from the users on the web is difficult task. Hence ***ranking*** is a good challenge for search engines. Moreover relevance of documents depends on user's nature and it is not standardized
- The concern of search engines is not only the ***content quantity*** but content quantity also.

- To avoid duplicate and near duplicate pages in the collection of web documents by search engines is a challenging task.
- Convert the user's complex queries into small and simple queries i.e. query formulation is also a challenge for search engines.
- To make the search personalized according to the user's interest is a big challenge.
- As the data on the web is unstructured, hence search becomes more complicated.
- To prevent the ***spamming*** on the web is a great challenge for any search engine.

1.2.3 Limitations of a Search Engines

Despite the fact that searching is incomplete without search engines; Search engines have their own limitations. Some of the limitations of search engines are as follows:

(1) Coverage problem

(2) Recall Precision problem

(3) Irrelevance problem

(4) Different search engines return different search results due to the variation in indexing and search process.

- **Coverage problem**

Ideally to find the relevant documents, any search engine must crawl the whole web but as web graph is very large in size, it is not possible to crawl the whole web graph fro any search engine. None of the search engines come close to indexing the entire web. Studies have shown that no search engine covers more than about 16% of the web and union of 11 major search engines covers less than 50% of the web Moreover, search engines are often out of date, partially due to limited crawling speed and the low average life span of web pages.[6]

- **Recall Precision problem**

Balance between precision and recall of query results is difficult to achieve. For broad queries, most of the general search engines generally return thousands of results thus yielding high recalls at the expenses of the precision web portals, where as yahoo, approaches the problem from the other extreme by organizing a very small subset of the web into a hierarchic structure which can yield high precision for a search but with low recall. [6]

- **Irrelevance problem**

Some SEs returns relevant web pages in response to a query based on keywords present in web pages. In that case they may miss many important pages. This can be understood by an example If we provide a query as "Toyota" which is a famous Japanese motor company, search engines my not return the home page of Toyota company because the home page of Toyota company does not possess Toyota keywords which may also lead to irrelevant results. Good quality search engines are trying to attempt this problem using link based methods. However developing good link based approaches is again a challenge.

- **Different search engines return different search results due to the variation in indexing and search process.**

The main reason of these all problems of search engines in that we do not have a good management of web data. Because web data explosion has taken the form of "Big Bang: and web data has no predefined uniform schema, it is a very tedious task to manage web data efficiently. And hence finding some useful and relevant information from the web according to user's query is a challenging and difficult task as compared to the data in conventional database management systems. These limitations of search engines have lead to development of meta search engines.

1.3 Meta Search Engine

A Meta Search engine is a system that supports unified access to multiple local search engines [2]. A Meta search engine sends user requests to several other search engines and or databases and aggregates the results into a single list or displays them according to their sources. MSEs reuse the indices of other services and do not store a local index.

Meta search engines works in various ways with simultaneous search engine sources, usually with the duplicates removed. Basically MSE's are spiders, which search the web and add meta tags to answer like Yahoo, Google etc. Some of the **examples of Some Meta Search Engines are:**

(1) Dog Pile MSE. (http://www.dogpile.com)

Popular Meta search site that sends a search a search to a customizable list of search engines, directories and specialty search sites, then displays results from each search engine individually.

(2) Meta Crawler (MC MSE)(http://www.metacrawler.com)

Meta crawler is one of the oldest Meta search engine. It is begun in July 1995 at the University of Washington. M c was purchased by info space, an online content provider, In Feb. 1997.

Some of the properties of meta search engines are as follows:

(1) Usually do not have their own database of crawled sites, They search the databases of real search engine

(2) MSE's gives you results by combining results from many search engines, say "yahoo" and "Google". However some MSEs do have database of sites submitted to them e.g. of MSE is DOGPILE.

(3) M SE can do additional preprocessing and post processing of results depending on need..

1.3.1 Advantages and Disadvantages of MSE's

Some of the advantages of MSEs are:

- Coverage is large as result of multiple search engines are combined
- Meta search engine have their own query language and hence the result does not depend upon syntax and Boolean operators of the individual search engines.
- Better quality of result: Retrieves the top ranking pages from the individual search engines.
- Post processing of result can be done such as clustering, better visualization etc.

Some of the disadvantages of meta search engines are :

- The unique feature of individual search engines is lost.
- **Not Exhaustive:** All the pages retrieved by each search engine are not displayed, only top ranking hits are considered.
- Each SE varies in quality, quantity, speed and other capabilities. Since a group of such search engines are accessed at a time the functionality of each SE affects the results.

- **Precision lost:** Most Meta search engines are using different SE's, they lose all advanced function of a traditional SE making precise search impossible.
- **Limited max of Results:** Most Meta search engine has limited number of results. This is due to the fact that those sites use other search engine which imposes a limit on use of their results.

1.3.2 Differences between Search Engine & Meta Search Engine

Search Engine

(1) A space, on the net, where one goes to find sites about specific information.

(2) Search engines usually have their own database of sites, which they craw and list in their database, e.g. Google, Yahoo.

(3) Search engine indexing collects, parses and store data to facilitate fast and accurate information retrieval

(4) Each search engine has its own ranking algorithm.

Meta Search Engine

(1) Basically MSEs has a "Spider robot" which searches the web and add the Meta tags to the answer like Google, yahoo etc.

(2) Usually do not have their own database of crawled sites, They search the databases of real search engines and combine their results to improve the quality of search

(3) MSEs reuse the indices of other services and do not store a local index.

(4) MSEs gives you results from multiple search engines say "yahoo" and "Google", however some MSEs do have database of sites submitted to them e.g. of MSE is DOGPILE.

1.3.3 Architecture of Meta Search Engine

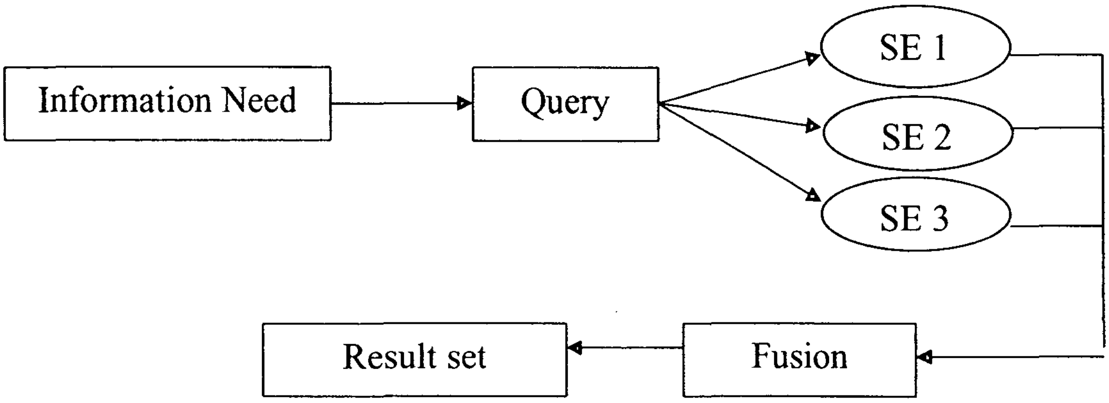

Fig: 1.2 Standard Architecture of Meta Search Engine. [4]

A standard meta-search engine accepts user's query and sends it to different search engines. The results obtained are fused to form a final result set. This standard architecture can be enhanced. Meta-search engine is a light way process therefore in a short time it can provide many other facilities to user. Such as Pre-processing and Post processing of the result. Following is architecture of a typical meta search engine:

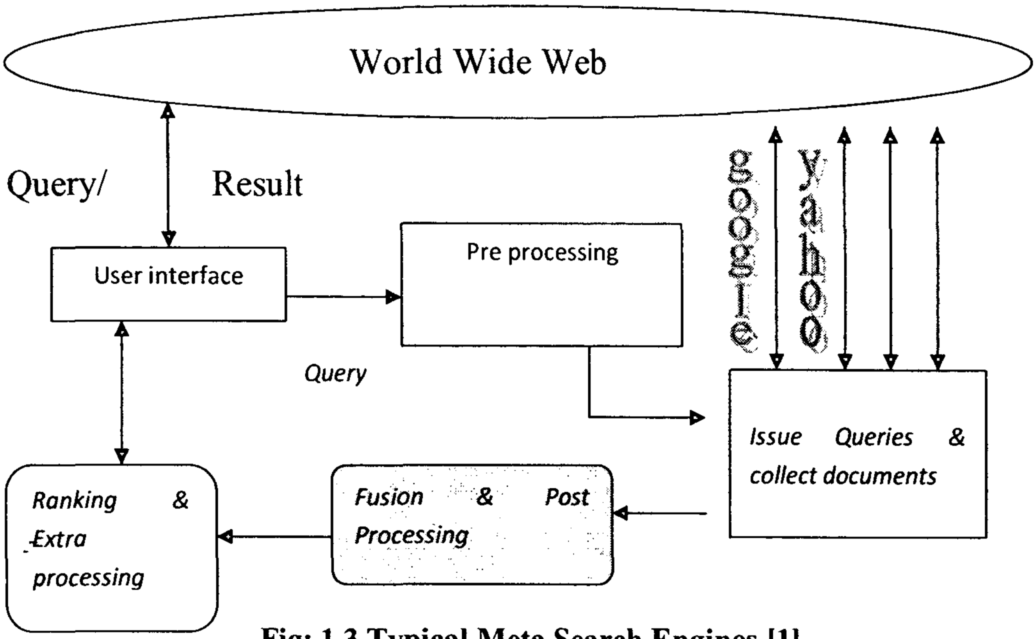

Fig: 1.3 Typical Meta Search Engines [1]

Following are the steps in working of MSE

(i) Accept a user query.
(ii) Process the query
(iii) Launch multiple queries.
(iv) Collect and merge results(collection fusion)
(v) Present post- processed results to the user.

User interface;

There are two important aspects of the user interface of Meta search engines

- *Query interface.* The basic query interface is a box where one or more words can be typed.
- *Answer interface.* The answer usually consist a list of the top ranking web pages.

Pre-Processing: Document pre-processing is a procedure which can be divided mainly into five text operations or transformation.

➢ Lexical analysis of the text with the objective of treating digits, hyphens, punctuation marks, and the case of letters.

➢ Elimination of stop words with the objective of filtering out words with very low discrimination values for retrieval purpose.

➢ Stemming of the remaining words with the objective of removing affixes(i.e. pre-fixes and suffixes) and allowing the retrieval of the documents containing syntactic variations of query terms (e.g. connect, connecting, connected, connection etc will be stemmed to give single word connect)

➢ Selecting of the index terms to determine which words/stems or groups of words will be used as an indexing element. In fact noun words frequently carry more semantics than adjectives, adverb, and verbs.

➢ Construction of term categorization structures.

Fusion &Post Processing: After getting document according to query, duplicate removal and advanced processing is done in this step with retrieved document.

Merging of results collected by MSE is called collection fusion, which combines retrieval results from multiple, independent collections into a single result such that effectiveness of searching entire set of documents as a single collection.

Ranking & Extra Processing: In this part of meta-search, simple or advanced re-ranking based on a variety of methods done and result is presented to user like any SE

1.3.4 Components of Meta Search Engine

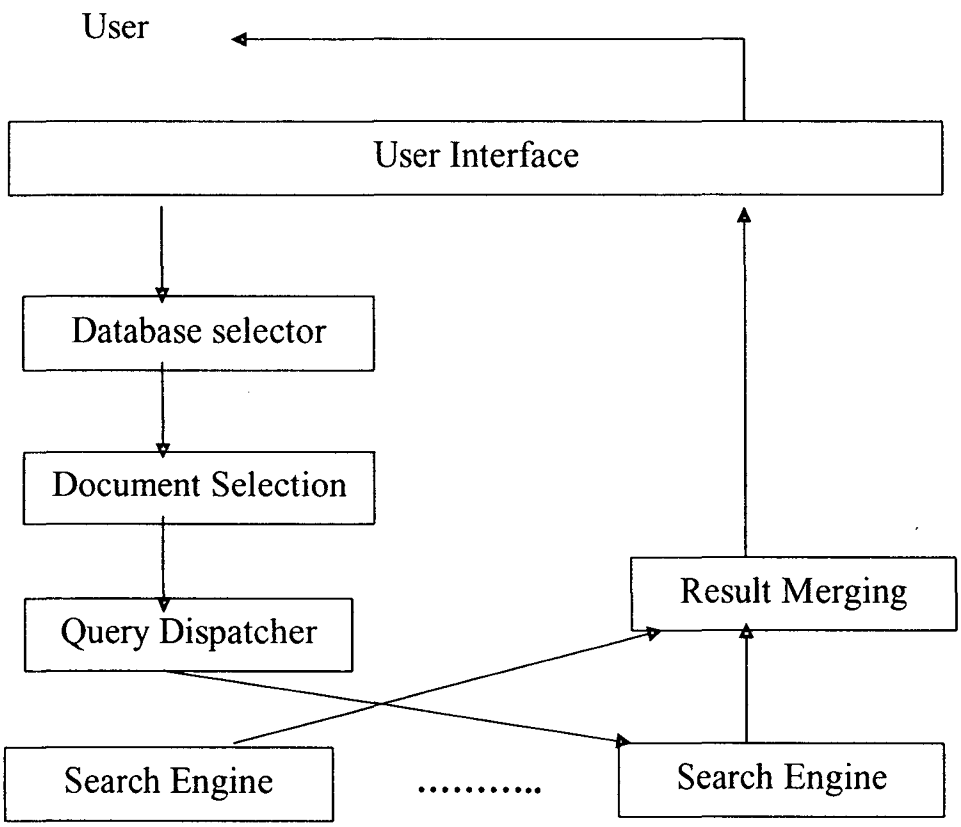

Fig: 1.4. Components of meta-search engine

Database Selection: Database selection is to identify search engines and it returns useful documents to a given query.

Document Selection: Document selection is to determine what document to retrieve from each identified search engine.

Query Dispatcher: Query dispatcher responsible for establishing a connection with the server of each SE and pass the query to it.

Result Merging: After dispatching the query the result will be merged from component SE and provide particular information to the user. The main complexity associated the merging process is determining how to re-rank the results in the presence and absence of their ranking scores assigned by their respective search engines.

CHAPTER 2 Database Selections and Document Selections

In this chapter I will focus on two important components of meta search engines dealing with database selection and document selection.

2.1 Database Selection:

Database Selection is a special component of Meta-search engine where a user submits a query to the Meta search engine through a user friendly interface. When database selector receives the query it sends it to different component search engines. Depending on the nature of query and user's need the query may be sent to a small specific or a large number of component search engines. For example, A user is interested in only the top 10 desired documents. These documents are contained in at most 10 search engine databases, however, if the no: of databases is much larger than 10, then large no: of databases will be useless w. r. t this query.
There are four problems while sending query to a large no: of component search engines.

First problem: Database wastes the resources at the MSE, when dispatching the query to useless DB's.
Second problem:
While transmitting data useless documents from SEs to MSEs would incur unnecessary network traffic.
Third problem:
When evaluating a query against useless component databases would waste resources at this local system.
Fourth problem:
If a large no: of documents were returned from useless databases, the more effort needed by the MSE to identify useful documents.
Therefore it is important to send each user query to only potentially useful databases. This means that a database selector should correctly identify for potentially useful databases while minimizing wrongly identifying useless databases as potentially useful ones.

2.2 Approaches for database selection

Many approaches have been proposed to tackle the database selection problem. In this subsection we discuss various approaches that can be used by database selector for identifying potentially useful databases depending on the user's need.

- **RRA (Rough Representative Approaches)**
- **SRA (Statistical Representative Approaches)**
- **LBA (Learning Based Approaches)**

2.2.1 RRA (Rough Representative Approaches)

In this approach contents of a local database are often represented by a few selected key words or paragraphs. This information only provide general idea on what a database is about, it is not very accurate in estimating the true usefulness of databases w.r.t given query.
Rough Representative approaches are often manually generated.

2.2.2 SRA (Statistical Representative Approaches)

In this approach, the content of a database are represented using detailed statistical information. The representative of a database contains some statistical information for each term in database such as the "document frequency" and the average weight of the term among all documents. It allow more accurate estimation of database usefulness w r t any query.
A large number of approaches based on statistical representative have been proposed. These are often three approaches in this approach.

1. **D-WISE Approach (Distributed WEB Index & Search Engine)**

2. **CORI-Net Approach (collection Retrieval Inference Network**

3. **gGloss Approach (generalized Glossary of Servers' Server)**

I will focus on D-WISE and CORI-Net.

2.2.2.1 D-WISE Approach (Distributed WEB Index & Search Engine)

D-WISE is proposed meta-search engine with a number of underlying search engines. The representative of a component search engine consists of the document frequency of each term in the component database as well as number of documents in the database. D-WISE uses the sum of weighted document frequencies of query terms. [16]

Therefore, the representative of a database with n distinct terms will contain n+1 quantity;
Let n_i denote the number of documents in the i_{th} components database & dfij be the documents frequency of term t_j in the i_{th} database.
Example: Suppose q is a user query the representatives of all databases are used to compute the ranking score of each component search engine with respect q. The scores measure the relative usefulness of all databases with respect to q. If the score of database A is higher than that database B, then database A will be more relevant to q then database B.
The ranking scores are computed as follows.
First, the "cue validity" of each query term, tj for the ith components databse, CVij, is computed using the following formula.

$$CVij = \frac{dfij \big/ ni}{\dfrac{dfij}{ni} + \dfrac{\sum_{k \neq i}^{N} df_{ki}}{\sum_{k \neq i}^{N} n_k}}$$

Where N is the total number component database in the meta-search engine, CVij measures the percentage of the documents in the ith database that contain term tj relative to the in all other databases. If the ith database has a higher percentage of documents containing tj in comparison to other databases then CVij leads to have a larger value. Next the "variance" of the CVij's of each query term tj for sll components databases, CCVj, is computed as follows:

$$CVVj = \frac{\sum_{i=1}^{N} (CVij - ACVj)^2}{N}$$

Where ACV is the average of all CVij's measures the skew of the distribution of term tj across all component databases. For two term tu and tv, if CVVu is a larger than CVVv, then term tu is more useful to distinguish different component databases than term tv. Finally the ranking score of component database I with respect to query q is computed by

$$r_i = \sum_{j=1}^{M} CVV_j \cdot df_{ij}$$

Where M is the number of terms in the query.

I= Ranking score of database, the sum of the document frequencies of all query term in the database weighted by each query term's CVV.

If the database has many useful query terms, each having a higher percentage of documents than other databases, then the ranking score of the database will be high. After the ranking scores of all databases are computed with respect to a given query, the database with the highest scores will be selected for search this query.

First, the ranking scores are relative scores. It will be difficult to determine that real value of a database with respect to a given query. If there are no good databases for a given query, then even the first ranked database will have very little value. If there are good database for another query, then 10th ranked database can be very useful.

Second, accuracy of this approach is questionable, as this approach does not take into consideration frequency of words in document. Thus it will treat a document with one occurrence of term, in same way as document containing 100 occurrence of the same term.

2.2.2.2 CORI-Net Approach (Collection retrieval Inference Network)

CORI-Net uses the probability that a database contains relevant documents due to the terms in a given query terms.

In this collection Retrieval Inference Network (CORI Net) Approach [9], the representative of a database consists of two pieces of information for each distinct term in the database: the document frequency and the database frequency.

If the term appears in multiple databases, only one databases frequency needs to be store in the meta-search engine to save space.

In CORI-Net, for a given query q, a documents ranking technique known as inference network [8] used in the INQUERY document retrieval system [27] is extended to rank all the component databases with respect to q. The extension is mostly conceptual and the main idea is to visualize the representatives of a database as a (super) document and the set of all representatives as a collection/database of super documents.

Consider as a super document containing all distinct terms in the database. If a term appears in k documents in the database, we repeat the term k time in the super document. As a result, the document frequency of a term in the database becomes the term frequency of the term in the super document. The set of all super documents of the component databases in the meta-search engine form a database of super documents.

In principle the *tf*w .*idf*w (term frequency weight times inverse document frequency weight) formula could now be used to compute the weight of each term in each super document so as to represent each supper document as a vector of weights.

A similarity function such as the cosine function may be used to compute the similarities (ranking scored) of all super documents (i.e. database representatives) with respect to query q and these similarities could then be used to rank all component databases.

In CORI-Net, the ranking score of a database with respect to query q is an estimated belief that the database contains useful documents.

Suppose the user query contains k terms $t_1 \ldots t_k$. Let N be the number of databases in the meta search engine. Let df_{ij} be the document frequency of the j-th term in the i-th component database D_i and dbf_i be the database frequency of the j-th term.

First, Di contains useful documents due to the j-th query term is computed by:

$$P\,(t_j \mid D_i) = c_1 + (1\text{-}c_1).\ T_{ij}.I_j \qquad (1)$$

Where

$$T_{ij} = c_2 + (1\text{-}c_2).\ \frac{df_{ij}}{df_{ij} \quad + \quad K}$$

Is the formula for computing the term frequency weight of the j-th term in the super document corresponding to Di and

$$I_j = \frac{\log\left(\frac{N+0.5}{db_j}\right)}{df_{ij}+k}$$

is the formula for computing the inverse document frequency weight of the j-th term based on all super documents . In the above formulas, c_1 and c_2 are constants between 0 and 1, and K= c_3. ((1-c_4) + c_4.dw_i/ad*w*) is a function of the size of database D_i with c_3 and c_4 being two constants, dw_i being the number of words in Di and *adw* being the average number of words in a database. The values of these constants (c_1, c_2, c_3, and c_4) can be determined empirically by performing experiments on actual test collections [].The value of p ($t_j \mid D_i$) is essentially the *tfw* . *idfw* weight of term t_j in the super document corresponding to database D*i*.

The significant of term t_j in representing query q, denoted p(q | t_j), can be estimated, for example, to be the query term weight of t_j in q. Finally that database D_i contains useful documents with respect to query q, or the ranking score of D_i with respect to q, can be estimated to be

$$r_i = p(q \mid D_i) = \sum_{j=1}^{k} p(q \mid t_j).\, p(t_j \mid D_i) \qquad (2)$$

In CORI-Net, the representative of a database contains slightly more than 1 piece of information per term (the document frequency plus the shared database frequency across all databases).Therefore, the CORI-Net approach also rather good scalability. The information for representing each component database can also be obtained and maintained easily.

An advantage of the CORI-Net approach is that the same method can be used to compute the ranking score of a a document with a query as well as the ranking score of a database (through the database representative or super document) with a query.

2.2.3 LBA (Learning Based Approaches)

In this approach, the usefulness of a database for new queries is based on the different retrieval results with the database from past queries. The retrieval result obtained in a three different of ways.

2.2.3.1 Static Learning Approach:

In Static learning approach, the retrieval knowledge of each component database with respect to training queries can be obtained in advanced (i.e. before the database selector is enabled). This type of approach is called *Static Learning Approach.*

The weakness of static learning approach is that it cannot be change the content of the database and query pattern.

2.2.3.2 Dynamic Learning Approach:

In Dynamic Learning Approach, real user queries (in contrast to training queries) can be used and the retrieval knowledge can be accumulated. It can be updated continuously. This type of approach is called *Dynamic Learning Approach.*

The weakness of this approach is that it may take a while to obtain sufficient knowledge useful to the database selector.

2.2.3.3 Combined Learning Approach:

In this approach, the static learning and dynamic learning can be combined to form a combined learning. In this approach initial knowledge may be obtained from training queries but the knowledge is updated continuously based on real user queries. The combined learning can overcome the weakness of learning approach. There are several learning based database selection methods

2.2.3.4 MRDD Approach (Modeling Relevant Document Distribution) Approach

The MRDD (Modeling Relevant Document Distribution) Approach [7] is a static learning approach. During learning, a set of training query is submitted to every component database. From the returned documents from a database for a given query, all relevant documents are identified and a vector reflecting the distribution of the relevant documents is obtained and stored.

EXAMPLE: 1

Consider a training query q and a component database D, 100 documents are retrieved of which d_1, d_4, $d_{10;}$ d_{17} and d_{30} (in order) are identified to be relevant. Then the distribution vector is $(r_1, r_2, r_3, r_4, r_5) = (1, 4, 10, 17, 30)$ where r_i is the positive integer indicating that r_i top ranked documents must be retrieved from database in order to obtain i relevant documents for the query. System finds such vector for all databases for all training queries. When user inputs a query, top k most similar training queries are obtained for each database. Average relevant document distributions over these k queries are obtained. This average distribution is used to find out the databases to be searched and documents to be retrieved. This selection tries to maximize the precision for each recall point.

EXAMPLE: 2

Suppose for a given query q, the following three average distribution vectors have been obtained for three component databases:

D_1 : (1, 4, 6, 7, 10, 12, 17)
D_2 : (3, 5, 7, 9, 15, 20)
D_3 : (2, 3, 6, 9, 11, 16)

Consider the case when three relevant documents are to be retrieved. To maximize the precision (i.e. to reduce the retrieval of irrelevant documents), one document should be retrieved from D1 and three documents should from D3 (two of the three are supposed to be relevant). The databases D1 and D3 should be selected. This selection yields retrieved of 0.75 as three out of the four retrieved documents are relevant.
In MRDD approach, the representative of a component database is the set of distribution vectors for all training queries.
The main weakness of this approach is that learning has to be carried out manually for each training query.

2.3 Document Selection

Document selection is very important and main part of the components database. In document selection we have to retrieve the document from each selected database. The naïve approach is that each component search engine returns all documents that are retrieved from search engine.
As the result, this approach will not only lead to higher communication cost but also require more effort from the result merger to identify the best matched documents. In this approach the problem is that too many documents may be retrieved from the component systems unnecessarily. Therefore naïve approach is not suitable for document selection.

The basic issue in document selection is to decide which documents to select form component database. The issue can be handled in two ways :

1. Determine the number of documents to retrieve from the component database
   If k documents are to be retrieved from a component database, then the k document with the largest local similarities will be retrieved.
2. Determine the local threshold for the components database such that a document from the component data base is retrieved only if its local similarity with the query exceeds the threshold.

Proposed approaches for the document selection problem can be divided into following categories.

- **User determination:**
- **Weighted allocation:**
- **Learning-based approaches:**
- **Guaranteed retrieval:**

2.3.1 User determination:

The meta-search engine lets the global user determine that how many documents to retrieve from each component database.

In Meta-Crawler [28, 29] and Savvy Search [30], the maximum number of documents to be returned from each component database can be customized by the user. Different numbers can be used for different queries. If the user does not select a number, then a query-independent default number set by the meta-search engine will be used. This approach may be reasonable if the number of component database is small and the user is reasonably familiar with all of them. In this case, the user can choose an appropriate number of documents to retrieve for each component database.

If the number of component databases is large, then this method has a serious problem. In this case, it is likely that the user will not be capable of selecting an appropriate number for each component database. Consequently, the user will be forced to choose one number and apply that number to all selected component databases. As the numbers of useful documents in different databases with respect to a given query are likely to be different, this method may retrieve too many useless documents from some component systems on one hand while retrieving too few useful documents from other component systems on the other hand. If *m* documents are to be retrieved from N selected databases, the number of documents to retrieve from each database may be set to be $\left\lceil \frac{m}{N} \right\rceil$ or slightly higher.

2.3.2 Weighted allocation:

The number of documents to retrieve from a component database depends on the ranking score (or the rank) of the component database relative to the ranking scores (or ranks) of other component databases that are retrieved from component databases that are ranked higher or have higher ranking scores.

In D-WISE [20], the ranking score information is used. For a given query q, let r_i be the ranking score of component database D_i, i=1... N, where N is the number of selected component databases for the query. Suppose m documents across all selected component databases are desired. Then the number of documents to retrieve from database D_i is $m .r_i / \sum_{j}^{N} r_j$.

In CORI-Net, the rank information is used. Specially, if a total number of *m* documents are to be retrieved from N component databases, then $m. \frac{2(1+N-i)}{N(N+1)}$ documents will be retrieved from the i-th ranked component database, i=1,..., N(note that $\sum_{i=1}^{N}\frac{2(1+N-i)}{N(N+1)}=1$). In CORI-Net, *m* could be chosen to be larger than the number of desired documents specified by the global user in order to reduce the like hood of missing useful documents.

As the special case of the weighted allocation approach, if the ranking score of a component database is the estimated number of potentially useful documents in the database, then the ranking score of a component database can be used as the number of documents to retrieve from the database.

Weighted Allocation is a reasonably flexible and easy to implement approach based on good intuition (i.e. retrieve more documents from more highly ranked local databases).

2.3.3 Learning-based approaches:

In this approach, the number of documents to retrieve from a component database based on past retrieval experiences with the component database.

A learning-based method, namely MRDD (Modeling Relevant Document Distribution), for database selection combines the selection of databases and the determination of what documents to be retrieved from databases. For given query q, after the average distribution vectors have been obtained for all databases, the decision on what documents to retrieve from these databases is made to maximize the overall precision.

Guaranteed retrieval:

This type of approach aims at guaranteeing the retrieval of all potentially useful documents with respect to any given query.

CHAPTER 3 Result Merging and Data Fusion

3.1 Introduction to Result Merging and Data Fusion

After dispatching the query, the result from component search engines has to be merged. Results merging problem is also known as data fusion Documents returned from each component search engine are ranked based on these documents' local ranking scores or similarities. However document in the merge result should be ranked in descending order of global similarities. An ideal merge is very hard to achieve due to the various heterogeneities among the component system search engine are ranked based on these documents.

The result merging approaches can be classified into the following two types.

(1) Local Similarity Adjustment
(2) Global similarity Estimation

3.2 Local Similarity Adjustment:

This approach merges the documents based on local similarity values provided by individual search engines. For this type of merging additional information measuring quality of results is required. Usually it is easier to implement but the merged ranking may be in accurate as the merge is not based on the true global similarities of returned documents. Some of the proposed functions to combine individual ranking scores include min, max, avg, sum weighted average and other linear combination functions

There are three cases for merging the results depending on the degree of overlap among the selected databases from a given query.

CASE.1 In this case, the databases are pair wise disjoint or nearly disjoint. This is generally the case when disjoint special purpose search engines or those with minimal overlap are selected.

CASE.2 In this case, the selected databases overlap but are not identical. For example when several general purposes search engines are selected.

CASE.3 In this case, the selected databases are identical.

Consider case first where all the returned documents are unique or minimum overlap. Each search engine has its own criteria for assigning local similarities and range of these similarity values is different. First step is to renormalize local similarity values in a uniform range. Actual weight of similarity values from particular component database depends on the importance of the database for specified query. During database selection process component databases are ranked according to their importance in satisfying user's need. Local similarity values are weighted using this information. A major problem with local adjustment is that local similarities of the returned documents from some components SEs are not available. Following approaches could be applied to tackle this problem.

(1) Use the local document rank information directly to perform the merge.

(2) Convert local document ranks to similarities.

1. Use the local document rank information directly type performs the merge.

In this approach, if the local similarities are available it will be ignored in this approach. The searched databases are arranged in descending order of usefulness or quality scores obtained during the database selection. A round robin method based on the database order and the local document rank order is used to merge the local document lists. The first document in the merged list is the top-ranked document from the highest-ranked database and the second document in the merged list is the top-ranked document in the merged list will be the second highest-ranked the database. After all searched databases have been selected the next document in the merged list will be the second highest-ranked document in the highest-ranked database and the process continues until the desired no: of documents are included in the merged list. The weakness of this solution is that it does not take into consideration the differences between the database score (i.e. only the order information is utilized)

2. Convert local document ranks to similarities.

As example, consider two databases D_1 & D_2. Suppose r_1=0.2 and r_2=0.5. Furthermore, suppose four documents are desired then, we have r_{min}=0.2, f_1=0.25, & f_2=0.1 based on the above conversion function, the top three ranked documents from D_1 will have converted similarities 1, 0.752, 0.5 respectively, and the top three ranked documents from D_2 will have converted similarities 1, 0.9 and 0.8, respectively. The merged list will contain three documents from D_2& one document from D_1. The documents will be ranked in descending order of converted similarities in the merged list.

3.3 Global similarity Estimation

Global similarity estimation is an ideal merging to compute true global similarities of returned documents.

Two methods can be applied for Global similarity Estimation.

- **Document Fetching**
- **Use of Discovered knowledge**

3.3.1 Document Fetching:

Typically the result of a search engine is collection of URLs of web documents along with some additional information about documents. The document fetching method downloads returned documents from their local servers and computes or estimates their global similarities in the meta-search engines. For finding global similarity MSE defines it own similarity function such as cosine to re-rank the documents returned by component search engines. For calculating global similarity meta-search engine requires document statistics of database and documents. After a document is downloaded, the term frequency of each term in the document can be obtained. As a result, all statistics needed to compute the global similarity of the document will be available and the global similarity can be computed.

3.3.2 Use of Discovered knowledge:

As we discussed previously, one difficulty with result merging is that local document similarities may be incomparable because in different component search engines the documents may be indexed differently and the similarities may be computed using different methods (term weighting schemes, similarity functions, etc.).If the specific document indexing and similarity computation methods used in different component search engines can be discovered, we can be in a better position to figure out

(1) What local similarities are reasonably comparable

(2) How to adjust some local similarities so that they will become more comparable with others

(3) How to derive global similarities from local similarities.

This is illustrated by the following example

Example: Suppose it is discovered that all the component search engines selected to answer a given user query employ the same methods to index local documents and to compute local similarities, and no collection-dependent statistics such as the *idf* information are used. Then the similarities from these local search engines can be consider as comparable. As a result, these similarities can be used directly to merge the returned documents.

If the only difference among these component search engines is that some remove stop words and some do not (or the stop word lists are different), then a query may be adjusted to generate more comparable local similarities.

Suppose a term t in query q is a stop word in component search engine E_1 but not a stop word in component search engine E_2. In order to generate more comparable similarities, we can remove t from q and submit the modified query to E_2 (it does not matter whether the original q or the modified q is submitted to E_1).

If the *idf* information is also used, then we need to either adjust the local similarities or computed the global similarities directly to overcome the problem that the global idf and the local *idfs* of a term may be different. Consider the following two cases. It is assumed that both the local similarity function and the global similarity function are the Cosine Function.

CASE 1: Query q consists of a single term t

The similarity of q with a document d in a component database can be computed by

$$sim(d,q) = \frac{qtf_t(q) \times lidf_t \times dtf_t(d)}{|q| \cdot |d|}$$

Where qtf_t(q) and dtf_t(d) are the *tf* weights of term *t* in *q* and in *d*, respectively, and $lidf_t$ is the local *idf* weight of *t*. If the local *idf* formula has been discovered and the global document frequency of *t* is known, then this local similarity can be adjusted to the global similarity by multiplying it by $\frac{gidf_t}{lidf_t}$, where $gidf_t$ is the global *idf* weight of *t*.

CASE 2: Query q has multiple terms $t_1 \ldots t_k$.

The global similarity between *d* and *q* in this case is

$$S = \frac{\sum_{i=1}^{k} qtf_{ti}(q) \times gidf_{ti} \times dtf_{ti}(d)}{|q| \cdot |d|}$$

$$S = \sum_{i=1}^{k} \frac{qtf_{ti}(q)}{|q|} . \frac{dtf_{ti}(d)}{|d|} . gidf_{ti}$$

Clearly, $\frac{qtf_{ti}(q)}{|q|}$ and $gidf_{ti}$ i=1...k,

Can all be computed by the meta-search engine as the formulas for computing them are known. Therefore in order to find 3, we need to find $\frac{dtf_{ti}(d)}{|d|}$, i=1...k. To find $\frac{dtf_{ti}(d)}{|d|}$ for a given term t_i without download document d, we can submit ti as a single-term query.

Let si = sim (d, ti) $= \frac{qtf_{ti}(t_i) \times lidft_i \times dtf_{ti}(d)}{|t_i| . |d|}$

Be the local similarity returned. Then

$$\frac{dtf_{ti}(d)}{|d|} = \frac{s_i \times |t_i|}{qtf_{ti}(d) \quad qtf_{ti}(t_i) \times lidf_{ti}}$$

The right hand side of the above formula can be computed by the meta-search engine when all the local formulas are known. In summary, *k* additional single-term queries can be computed the global similarities between *q* and all documents retrieved by *q*.

3.4 Some Popular Approaches for Data Fusion

In general, a data fusion algorithm accepts two or more rank list and merges these lists into single ranked list with the aim of providing a better effectiveness than all systems used for data fusion [33].

Data fusion in automatic evaluation determines the (pseudo) relevant documents for evaluating the retrieve performance of a set of retrieval systems. For this purpose, the retrieval results of the systems to be ranked are merged following various techniques and the top ranked documents in the merged results are considered as "PSEUDORELevant DocumentS" (pseudorels) and used to evaluate the relative effectiveness of retrieval systems. We refer these documents as "(pseudo) relevant documents".

The meta-search (data fusion) software involves four components:

1. Database/search engine selector: the search engines (databases) to be fused selected using system selection methods.
2. Query dispatcher: the queries are submitted to the underlying search engines.
3. Document selector: documents to be used from each search engine are determined. The simplest way is the use of the top b documents.
4. Result merger: the results of search engines are merged using some merging techniques.

Data fusion methods for determining *pseudorels*

There are three data fusion methods for determining the *Pseudorels: the Rank Position, Borda Count, and Condorcet methods.*

3.4.1 Rank Position (Reciprocal Ranking) method

In this approach, to merge the documents into a unified list only the rank positions of retrieved documents are used. Retrieval systems determine the rank positions .When a duplicated document is found the inverse of its rankings are summed up, since the documents returned by more than on retrieval system might be more likely to be relevant. The following equation shows the computation of the rank score of document *I* using the position information of this document in all of the systems (j=1...n).

$$r(d_i) = \frac{1}{\sum_j \frac{1}{position(d_{ij})}}$$

Note that in this summation, systems not ranking a document are skipped.

In this approach, first Rank Position score of each document to be combined is evaluated, then using these rank position scores, documents are sorted in non-decreasing order. A portion (e.g., a certain percentage) of the top documents is treated as pseudorels.

Example: Suppose that we have four different retrieval systems A, B, C, and D with a documents collection composed of documents a, b, c, d, e, f, and g. for a given query their top four results are ranked as follows:

A = (a, b, c, d)
B = (a, d, b, e)

C = (c, a, f, e)
D = (b, g, e, f)

Now, we can compute the rank position of each document in the document list, and the rank scores of the documents are as follows:

r (a) = 1/ (1+1+1/2) = 0.4
r (b) = 1/ (1/2+1/3+1) = 0.54, and so on

The final ranked list of documents is a>b>c>d>e>f>g, i.e., *a* is the document with the highest rank, i.e., it is the top most document; b is the second document, etc.

3.4.2 Borda Ranking

The first method taken from social theory of voting and used in the data fusion is Borda count method, which is introduced by Jeans-Charles de Borda count.

The highest ranked individual (in an n- way vote) get n votes and each subsequent gets one vote less (so the number two gets n-1 and the number three gets n-2 and so on). If there are candidates left unranked by the voter, the remaining points are divided evenly among the unranked candidates. Then for each alternative, all the votes are added up and the alternative with highest number of votes wins the election.

Example:

Suppose that we have three retrieval systems *A, B,* and *C* and *a, b, c, d* and *e* are pages/documents to be ranked. Retrieval systems give following ranked list of documents:

A = (a, c, b, d)
B = (b, c, a, e)
C = (c, a, b, e)

The Borda Count (BC) of each document is computed by summing their Borda count values in individual systems as follows:

$BC\ (a) = BC_A\ (a) + BC_B\ (a) + BC_C\ (a) = 5+3+4 = 12$
$BC\ (b) = BC_A\ (b) + BC_B\ (b) + BC_C\ (b) = 3+ 5+3 = 11$

Finally, the documents are ranked using their Borda counts. The final ranked list of documents is c > a > b > e > d.

In Borda Count, the deletion of a document may reverse the rank positions of other documents. We ignore such cases (if any). For example, some Web search engines, such as Google, use undisclosed algorithms that exploit page linking information among Web pages for final ranking of response URL's; deletion of page may change the association among the remaining pages and reverse the rankings of some documents.

3.4.3 Condorcet Ranking

The second method from social theory of voting, Condorcet's algorithm, is named after the French mathematician *Marie Jean Antoine Nicolas de Caritat Condorcet.* In the Condorcet election method, voters rank the candidates in the order of preference. The vote counting procedure then takes into account each preference of each voter for one candidate over another. Then Condorcet voting algorithm is a majoritarian method that specifies the winner as the candidate, which beats each of the other candidates in a pair wise comparison.

Example: Suppose that we have three candidates (documents) a, b, and c with five voters (systems) A, B, C, D, and E. (Note that in system C, the documents b and c have the same original rank.)

A: a > b > c
B: a > c > b
C: a > b = c
D: b > a
E: c > a

In the first stage, we use an N · N matrix for the pair wise comparison, where N is the number of candidates. Each non-diagonal entry *(i, j)* of the matrix shows the number of votes *i* over *j* (i.e., cell [*a,b*] shows the number of wins, loses, and ties of document a over document b, respectively). In a system while counting votes, a document loses to all other retrieved documents if it is not retrieved by that system.

	a	b	c
a	-	4, 1, 0	4, 1, 0
b	1, 4, 0	-	2, 2 1
c	1, 4, 0	2, 2, 1	-

After that, we determine the pair wise winners. Each complimentary pair is compared, and the winner receives one point in it's ''win'' column and the loser receives one point in it's ''lose'' column. If the simulated pair wise election is a tie, both receive one point in the ''tie'' column.

	Win	Lose	Tie
a	2	0	0
b	0	1	1
c	0	1	1

To rank the documents we use their win and lose values. If the number of wins that a document has is higher than the other one, then that document wins. Otherwise if their win property is equal we consider their lose scores; the document which has smaller lose score wins. If both win and lose scores are equal then both documents are tied. The final ranking of the documents in the example is a > b = c.

CHAPTER 4 Experiments on Data Fusion for Ranking System

4.1 Data fusion for ranking systems

A data fusion algorithm accepts two or more ranked lists and merges these lists into a single ranked list with the aim of providing a better effectiveness than all individual systems used for data fusion [33]. Some of the popular data fusion algorithms are given in Chapter 3 in section 3.4.

4.2 Experiments and Results

Objective: To merge ranked list of web pages obtained from individual search engines.

Input:

1. User query
2. ranked list of web pages provided by different search engines for the specified query

Output: a final ranked list of web pages

Sample Input

Query: tourist places in India

Web pages: Results obtained from Yahoo, MSN and Google (See table 4.1-4.3)

Steps in Experiments

Step 1: In step1, we take a query to find out the results from different databases or search engine, such as Yahoo, Google and Msn

Step 2: In step 2, we consider top k documents from individual search engines

Step 3: In step 3, we find out union of top k documents obtained from individual engines. So we will remove the duplicate and get the unique results from different search engines in order to get unique web pages.

Step 4: We apply the formula for merging the results of individual search engines to get the final ranked result. (We have used Reciprocal ranking and Borda ranking in our experiment)

Experiments were performed for two methods: Reciprocal Ranking and Borda rank. For both the methods sample queries (50 queries) were chosen, top 20 results each from Yahoo, Google and MSN search engines were taken. The results obtained by individual search engines were merged using the specified method. Quality of result was checked by observing the results and finding out which method was able to fulfill user's need in more efficient way. We are presenting results of one of the queries using the two methods as discussed above.

Intermediate Result after Step 1

Results after step 1

For k=20 and query ="tourist places in India", top k results by Yahoo, Google and Msn search engine are shown in tables 4.1, 4.2, and 4.3.

Table 4.1: Top 20 results for query "Tourist places in India" by Yahoo search engine

S.N	URL's
1	www.touristplacesinindia.com
2	http://wwwtouristplacesofindiacom
3	http://wwwtouristplacesinindiacom/hill-stations-of-indiahtml
4	http://indiantouristplacesinfo
5	http://wwwtouristic-places-indiacom/indexhtml
6	http://wwwindia-tourist-placescom
7	http://wwwparadiseindiacom/TouristPlaces
8	http://wwwmustseeindiacom/tourist-places-in-india
9	http://wwwmustseeindiacom/tourist-places-in-india
10	http://wwwindianholidaycom/tourist-attractions

11	http://wwwrrindiacom/tourist-attractionshtml
12	wwwprokeralacom/maps/india/india-tourist-places
13	http://wwwfamous-indiacom
14	wwwtraveltoindianet/tourist-pick
15	wwwtouristplacesofindiacom/sitemap
16	http://wwwsouthindiatourtravelcom/karnataka/tourist-places/indexhtml
17	http://wwwtouring-indiacom
18	http://wwwtouristspotsindiacom/indexhtml
19	http://wwwtripadvisorin/SmartDeals-g293860-India-Hotel-Dealshtml
20	http://wwwindianluxurytourscom/tourist-attractions/

Table 4.2: Top 20 results for query "Tourist places in India" by Google search engine

S.N	URL's
1	www.touristplacesinindia.com
2	http://wwwtouristplacesinindiacom/hill-stations-of-indiahtml
3	http://wwwindiaplacescom
4	http://wwwtouring-indiacom
5	http://wwwtoptouristplacescom
6	http://wwwmapsofindiacom/maps/india/tourist-centershtm
7	http://wwwtouristplacesofindiacom
8	http://wwwfamous-indiacom
9	http://wwwdestinationindiatoursincentivescom
10	http://wwwmapsofindiaorg/india-tourist-attractions-pictures/indexhtml

11	http://wwwi-indiaonlinecom/prog_ladlihtm?gclid=CMyVppPd_qICFUpB6wodL1Fiaw
12	http://wwwsaffronsofindiacom/wwwtripadvisorin/SmartDeals-g293860-India-Hotel-Dealshtmhttp://wwwtouristic-places-indiacom
13	http://wwwincredibleindiaorg/indexhtml
14	http://wwwindiatouristspotscom/
15	http://wwwsouthindiatourtravelcom/
16	http://wwwindiaprofilecom/discover-india
17	http://wwwindianholidaycom/tourist-attractions
18	httphttp://enwikipediaorg/wiki/Tourism_in_India
19	http://wwwtrainenquirycom/staticcontent/tourist_info/home1html
20	http://www.dailytimes.com.pk/default.asp?page=2010%5C04%5C04%5Cstory_4-4-2010_pg3_2

Table 4.3: Top 20 results for query "Tourist places in India" by MSN search engine

S.N	URL's
1	www.touristplacesinindia.com
2	www.ouristplacesofindia.com
3	Wwwindiaplacescom
4	wwwtouristplacesinindiacom/tourist-places-indiahtml
5	wwwmapsofindiacom/maps/india/tourist-centershtm
6	http://wwwindianholidaycom/tourist-attractions
7	http://wwwfamous-indiacom
8	http://indiatouristplaces4ublogspotcom

9	http://wwwtourist-places-indiacom
10	http://wwwtoptouristplacescom
11	http://wwwindiatoursorgin
12	http://trainenquirycom/StaticContent/Tourist_Info/home1html
13	http://tourismindiasitescoin
14	http://wwwmaharashtratourismgovin
15	http://wwwtourismofkeralacom/destinations/indexhtml
16	http://enwikipediaorg/wiki/Tourism_in_India
17	http://wwwindiatouristplacecom/
18	http://wwwindianholidaycom/tourist-attractions
19	http://touring-indiacom
20	http://touristic-places-indiacom

Result (Method 1)

In this method Reciprocal ranking was used to merge the Web pages. Result obtained is as follows:

Rank Position (Reciprocal Ranking)

URL's NO (Sorted in ranked order)*	Rank of yahoo	Rank of Google	Rank of MSN	Highest Engine	Score for Reciprocal Ranking
1	1	1	1	Yahoo	0.33
2	3	2	0	Google	1.20
3	2	6	0	Yahoo	1.50
4	0	0	2	MSN	2.00

5	13	7	7	Google	2.76
6	0	3	0	Google	3.00
7	0	0	3	MSN	3.00
8	10	17	6	MSN	3.07
9	17	4	0	Google	3.24
10	0	5	10	Google	3.33
11	0	0	4	MSN	4.00
12	4	0	0	Yahoo	4.00
13	0	0	5	MSN	5.00
14	5	0	0	Yahoo	5.00
15	0	5	0	Google	5.00
16	6	0	0	Yahoo	6.00
17	7	0	0	Yahoo	7.00
18	0	8	0	Google	8.00
19	8	0	0	Yahoo	8.00
20	0	0	8	MSN	8.00
21	0	0	9	MSN	9.00
22	0	9	0	Google	9.00
23	10	0	0	Google	10.00
24	11	0	0	Yahoo	11.00
25	0	0	11	MSN	11.00
26	0	11	0	Google	11.00
27	0	12	0	Google	12.00

28	0	0	12	MSN	12.00
29	12	0	0	Yahoo	12.00
30	0	13	0	Google	13.00
31	0	0	13	MSN	13.00
32	0	0	14	MSN	14.00
33	14	0	0	Yahoo	14.00
34	0	14	0	Google	14.00
35	0	0	15	MSN	15.00
36	0	15	0	Google	15.00
37	15	0	0	Yahoo	15.00
38	0	0	16	MSN	16.00
39	0	16	0	Google	16.00
40	16	0	0	Yahoo	16.00
41	0	0	17	MSN	17.00
42	18	0	0	Yahoo	18.00
43	0	18	0	Google	18.00
44	0	19	0	Google	19.00
45	19	0	0	Yahoo	19.00
46	0	0	19	MSN	19.00
47	0	20	0	Google	20.00
48	20	0	0	Yahoo	20.00
49	0	0	20	MSN	20.00

***See Appendix 1 Table 1 for list of URLs**

Result (Method 2)

In this method Borda ranking was used to merge the Web pages. Result obtained is as follows:

Borda Ranking

URL's NO (Sorted in ranked order) *	Rank of yahoo	Rank of Google	Rank of MSN	Highest Engine	Score for Borda Ranking
1	1	1	1	Yahoo	147.00
2	13	7	6	Google	123.00
3	10	17	6	MSN	117.00
4	3	2	0	Google	110.00
5	2	6	0	Yahoo	107.00
6	0	5	10	Google	100.00
7	17	4	0	Google	94.00
8	0	0	2	MSN	77.50
9	0	3	0	Google	77.00
10	0	0	3	MSN	76.50
11	4	0	0	Yahoo	75.50
12	0	0	4	MSN	75.50
13	0	5	0	Google	75.00
14	5	0	0	Yahoo	74.50
15	0	0	5	MSN	74.50
16	6	0	0	Yahoo	73.50

17	7	0	0	Yahoo	72.50
18	0	8	0	Google	72.00
19	0	0	8	MSN	71.50
20	8	0	0	Yahoo	71.50
21	0	9	0	Google	71.00
22	0	0	9	MSN	70.50
23	0	10	0	Google	70.00
24	0	11	0	Google	69.00
25	0	0	11	MSN	68.50
26	11	0	0	Yahoo	68.50
27	0	12	0	Google	68.00
28	12	0	0	Yahoo	67.50
29	0	0	12	MSN	67.50
30	0	13	0	Google	67.00
31	0	0	13	MSN	66.50
32	0	14	0	Google	66.00
33	14	0	0	Yahoo	65.50
34	0	0	14	MSN	65.50
35	0	15	0	Google	65.00
36	15	0	0	Yahoo	64.50
37	0	0	15	MSN	64.50
38	0	16	0	Google	64.00
39	0	0	16	MSN	63.50

40	16	0	0	Yahoo	63.50
41	0	0	17	MSN	62.50
42	0	18	0	Google	62.00
43	18	0	0	Yahoo	61.50
44	0	19	0	Google	61.00
45	19	0	0	Yahoo	60.50
46	0	0	19	MSN	60.50
47	0	20	0	Google	60.00
48	0	0	20	MSN	59.50
49	20	0	0	Yahoo	59.50

***See Appendix 1 Table 2 for list of URLs**

Result Analysis

Observing the result I see that there is an overlapping of just about 10 % in the results of individual search engines (Yahoo, Google and MSN). I had presented the result for a query (tourist places in India), where I expected a larger overlap as I think that query is focused in the sense that user's needs are well defined and also the result is expected to be well defined. But the result is not as expected. This variation in the results generated by individual search engines motivates one to combine the result of various search engines, so as to improve retrieval efficiency of the results. For less focused queries, the overlapping may still be less and it may be a quite challenging task to select relevant documents appropriately. As discussed above there are different approaches of merging and fusing the result, so user has to be selective in choosing appropriate approach so as to get maximum retrieval efficiency. In our case I find that Reciprocal ranking is giving better result. This was true for most of the queries on which experiments were performed. The efficiency of two approaches was calculating by observing the results and finding, which method fulfills user's need in better way. Extensive theoretical as was as practical study is required in order to evaluate effectiveness of various approaches for document selection.

Appendix 1 Table:no.4.6

*Url's are given in appendix no.

Table: no.1

S.NO: URL's	Name of the web pages
1	Wwwtouristplacesinindiacom
2	http://wwwtouristplacesinindiacom/hill-stations-of-indiahtml
3	http://wwwtouristplacesofindiacom
4	Wwwtouristplacesofindiacom
5	http://wwwfamous-indiacom
6	http://wwwindiaplacescom/
7	Wwwindiaplacescom
8	http://wwwindianholidaycom/tourist-attractions/
9	http://wwwtouring-indiacom
10	http://wwwtoptouristplacescom
11	wwwtouristplacesinindiacom/tourist-places-indiahtml
12	http://indiantouristplacesinfo/
13	wwwmapsofindiacom/maps/india/tourist-centershtm
14	http://wwwtouristic-places-indiacom/indexhtml
15	http://wwwmapsofindiacom/maps/india/tourist-centershtm
16	http://wwwindia-tourist-placescom
17	http://wwwparadiseindiacom/TouristPlaces
18	http://wwwdestinationindiatoursincentivescom

19	http://wwwmustseeindiacom/tourist-places-in-india
20	http://indiatouristplaces4ublogspotcom
21	http://wwwtourist-places-indiacom
22	http://wwwmapsofindiaorg/india-tourist-attractions-pictures/indexhtml
23	http://wwwi-indiaonlinecom/prog_ladlihtm?gclid=CMyVppPd_qICFUpB6wodL1Fiaw
24	http://wwwrrindiacom/tourist-attractionshtml
25	http://wwwindiatoursorgin
26	http://wwwsaffronsofindiacom/wwwtripadvisorin/SmartDeals-g293860-India-Hotel-Dealshtmhttp://wwwtouristic-places-indiacom/l
27	http://wwwtouristic-places-indiacom
28	http://trainenquirycom/StaticContent/Tourist_Info/home1html
29	wwwprokeralacom/maps/india/india-tourist-places
30	http://wwwincredibleindiaorg/indexhtml
31	http://tourismindiasitescoin
32	http://wwwmaharashtratourismgovin
33	wwwtraveltoindianet/tourist-pick
34	http://wwwindiatouristspotscom
35	http://wwwtourismofkeralacom/destinations/indexhtml
36	http://wwwsouthindiatourtravelcom
37	wwwtouristplacesofindiacom/sitemap
38	http://enwikipediaorg/wiki/Tourism_in_India
39	http://wwwindiaprofilecom/discover-india

40	http://wwwsouthindiatourtravelcom/karnataka/tourist-places/indexhtml
41	http://wwwindiatouristplacecom
42	http://wwwtouristspotsindiacom/indexhtml
43	httphttp://enwikipediaorg/wiki/Tourism_in_India
44	http://wwwtrainenquirycom/staticcontent/tourist_info/home1html
45	http://wwwtripadvisorin/SmartDeals-g293860-India-Hotel-Dealshtml
46	http://touring-indiacom
47	http://www.dailytimes.com.pk/default.asp?page=2010%5C04%5C04%5Cstory_4-4-2010_pg3_2
48	http://wwwindianluxurytourscom/tourist-attractions
49	http://touristic-places-indiacom

Table: no 2.

S.NO: URL's	Name of the web pages
1	wwwtouristplacesinindiacom
2	http://wwwfamous-indiacom
3	http://wwwindianholidaycom/tourist-attractions
4	http://wwwtouristplacesinindiacom/hill-stations-of-indiahtml
5	http://wwwtouristplacesofindiacom
6	http://wwwtoptouristplacescom
7	http://wwwtouring-indiacom/
8	Wwwtouristplacesofindiacom

9	http://wwwindiaplacescom/
10	Wwwindiaplacescom
11	http://indiantouristplacesinfo
12	wwwtouristplacesinindiacom/tourist-places-indiahtml
13	http://wwwmapsofindiacom/maps/india/tourist-centershtm
14	http://wwwtouristic-places-indiacom/indexhtml
15	wwwmapsofindiacom/maps/india/tourist-centershtm
16	http://wwwindia-tourist-placescom
17	http://wwwparadiseindiacom/TouristPlaces
18	http://wwwdestinationindiatoursincentivescom
19	http://indiatouristplaces4ublogspotcom
20	http://wwwmustseeindiacom/tourist-places-in-india
21	http://wwwmapsofindiaorg/india-tourist-attractions-pictures/indexhtml
22	http://wwwtourist-places-indiacom
23	http://wwwi-indiaonlinecom/prog_ladlihtm?gclid=CMyVppPd_qICFUpB6wodL1Fiaw
24	http://wwwsaffronsofindiacom/wwwtripadvisorin/SmartDeals-g293860-India-Hotel-Dealshtmhttp://wwwtouristic-places-indiacom/1
25	http://wwwindiatoursorgin
26	http://wwwrrindiacom/tourist-attractionshtml
27	http://wwwtouristic-places-indiacom
28	wwwprokeralacom/maps/india/india-tourist-places
29	http://trainenquirycom/StaticContent/Tourist_Info/home1html

30	http://wwwincredibleindiaorg/indexhtml
31	http://tourismindiasitescoin/
32	http://wwwindiatouristspotscom
33	wwwtraveltoindianet/tourist-pick
34	http://wwwmaharashtratourismgovin
35	http://wwwsouthindiatourtravelcom
36	wwwtouristplacesofindiacom/sitemap
37	http://wwwtourismofkeralacom/destinations/indexhtml
38	http://wwwindiaprofilecom/discover-india/
39	http://enwikipediaorg/wiki/Tourism_in_India
40	http://wwwsouthindiatourtravelcom/karnataka/tourist-places/indexhtml
41	http://wwwindiatouristplacecom
42	http://enwikipediaorg/wiki/Tourism_in_India
43	http://wwwtouristspotsindiacom/indexhtml
44	http://wwwtrainenquirycom/staticcontent/tourist_info/home1html
45	http://wwwtripadvisorin/SmartDeals-g293860-India-Hotel-Dealshtml
46	http://touring-indiacom
47	http://www.dailytimes.com.pk/default.asp?page=2010%5C04%5C04%5Cstory_4-4-2010_pg3_2
48	http://touristic-places-indiacom
49	http://wwwindianluxurytourscom/tourist-attractions

CHAPTER 5 Conclusions

Meta search engine is system that supports unified access to multiple local search engines. Our overview concentrated on the problems of database selection, document selection and result merging. I have also discussed the causes that make these problems very challenging. My main focus was on studying various techniques in database selection, document selection and result merging to build the efficient and effective meta-search system.

In this dissertation, I report my study on how to merge the search results returned from multiple search engines into the ranked list. This is an important problem in meta-search engine research. An effective and efficient result merging strategy is essential for developing effective meta-search system. I have experimented on automatic ranking of retrieval systems without relevance judgments using two different data fusion techniques: the Rank Position (Reciprocal Ranking) and Borda Ranking methods. I have compared the effectiveness of these two methods in automatic ranking.

References

[1] Manoj. M and Elizebeth Jacob, Information retrieval on the internet using Meta search engines, A review journal of scientific & industrial research vol 67, oct 2008, pp739-746

[2] Zonghuan Wu, Weiyi Meng, Clement Yu, Zhuogang Li, Towards a Highly-scalable and effective meta search engine, Proceedings of the 10th international conference on World Wide Web, p.386-395, May 01-05, 2001

[3] http:/www.allmetasearch.com

[4] Eric J. Glover , Steve Lawrence , William P. Birmingham , C. Lee Giles, Architecture of a meta search engine that supports user information needs, Proceedings of the eighth international conference on Information and knowledge management, p.210-216, November 02-06, 1999,

[5] W.Meng,C.Yu,K.Liu. Building Efficient and Effective Meta Search Engines. ACM computing surveys, 34(1), March 2002, pp.48-84

[6] G. W.Flake, S.Lawrence and C.L.Giles, "Efficient Identification of Web Communities", In Proc. Of 6th ACM SIGKDD Conference on Knowledge discovery and data mining, Boston, MA, USA, pp.150-160, 2000.

[7] VOORHEES, E., GUPTA, N., AND JOHNSON-LAIRD, B. 1995b.Learning collection fusion strategies. In proceedings of the ACM SIGIR conference (Seattle, WA, July 1995), 172-179

[8] CALLAN, J., CROFT, B., and HARDING, S. 1992.The inquiry retrieval system. In Proceedings of the Third DEXA conference (Valencia, Spain, 1992), p.78-83,

[9] CALLAN, J., Lu, Z., and CROFT, W. 1995. Searching distributed collections with inference networks. In Proceedings of the ACM SIGIR conference (Seattle, WA, July 1995),-21-28.

[10] KOSTER, M.1994. Aliweb: Archie-like indexing in the web. Computer network and ISDN system. 27, 2, 175-182.

[11] Web Communities Analysis and Construction Springer by Yanchun Zhang. Jeffrey XuYu. Jingyu Hou, ACM computing classification (1998)

[12] Information Retrieval Algorithms and Heuristics. 1998 by Kluwer Acadmic Publishers.

[13] Yu, C., Meng, W., Wu, W., and Liu, K.2001. Efficient and effective meta-search for text databases incorporating linking among documents. In Proceedings of the ACM SIGMOD Conference, Santa Barbara, CA (May 2001), pp. 187-198.

[14] David A. Grossman phir Frieder Information Retrieval algorithms and Heuristics 2008, pp.2-224

[15] Souman Chankrabarti Mining the web, Discovery knowledge from Hypertext data 2002/10/09,pp.2-334.

[16] B.Yuwono, and D. lee, Server Ranking for Distributed Text Resource system on Internet. DASFA A' 97, 1997.

[17] L.Yi, B. Lui, and X. Li, "Eliminating Noisy Information in web pages for Data Mining", in proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD-2003), Washington DC, USA, 2003.

[18] C. Yu, and W.Meng. Principles of database Query processing for advanced Applications. Morgan Kaufmann Publishers, San Francisco, 1998.

[19] C. Yu, W. Meng, K. Liu, W. Wu, and N. Rishe. Efficient and effective meta-search for a large Number of Text databases. CIKM'99, 1999.

[20] B. Yuwono, and D. Lee. Server Ranking for Distributed Text Resource Systems on the Internet. In Proceedings of the 5th International Conference on Database Systems for Advanced Applications, Melbourne, Australia (April 1997), pp.391-400.

[21] Brian D. Davison. The potential of the meta-search engine. In proceedings of the annual Meeting of the American Society for Information Science and Technology, providence, RI, November 2004

[22] Lawrence, S. And Lee Giles, C. 1999. Accessibility of information on the web. Nature 400, 107-109.

[23] Bergman, M.2000. The deep Web: Surfacing the hidden value. Bright Planet, www.completeplanet.com/Tutorial/DeepWeb/index.asp

[24] J.Aslam, M. Montague. Models for Meta-search. ACM SIGIR conference, 2001, pp.276-284.

[25] C.Dwork, R.Kumar, M.Naor, D.Sivakumar. Rank Aggregation Methods for the web. Tenth International World Wide Web Conference, pp. 613-622, 2001.

[26] Flake, G.W, Lawrence, S., & Giles, C. L. (2000). Efficient identification of web communities. In proceedings of the 6th ACM International conference on knowledge Discovery and Data Mining (SIGKDD-2000), pages 150-160, Boston.

[27] Callan,J.Lu,Z., and Croft, W.1995. Searching distributed collections with inference networks. In Proceedings of the ACM SIGIR conference, Seattle (July 1995), pp.21-28.

[28] Selberg, E. and Etzioni, O. 1995. Multi-service search and comparison using the meta-crawler. In Proceedings of the Fourth World Wide Web Conference, Boston, Massachusetts (December 1995), pp.195-208.

[29] Selberg, E and Etzioni, O. 1997. The meta-crawler architecture for resource aggregation on the web. IEEE Expert 12, 1, 8-14.

[30] Dreilinger, D. And Howe, A. 1997. Experiences with selecting search engine using meta search.ACM Transactions on information systems 15, 3 (July), 195-222.

[31] Liu, K., Meng, W., Yu, C., and Rishe, N.2000. Discovery of similarity computations of search engines. In Proceedings of the 9th ACM International Conference on Information and Knowledge Management, Washington, D.C (November 2000), pp.290-297.

[32] Meng, W., Yu, C., And Liu, K.1999b. Detection of heterogeneities in a multiple text database environment. In Proceedings of the Fourth IFCIS Conference on Cooperative Information Systems, Edinburgh, Scotland (September 1999), pp.22-33.

[33] Croft, W. 2000. Combining approaches to information retrieval. In Advances in Information Retrieval: Recent Research from the Center for Intelligent Information Retrieval, edited by W. Bruce Croft. Kluwer Academic Publishers (2000), pp.1-36.

[34] Can, F., Nuray, R., & Sevdik, A. B. (2004). Automatic performance evaluation of Web search engines. Information Processing and Management, 40(3), 495–514.

[35] Nuray, R., & Can, F. (2003). Automatic ranking of retrieval systems in imperfect environments. In Proceedings of the 26th ACM SIGIR conference (pp. 379–380).

[36] Wu, S., & Crestani, F. (2003). Methods for ranking information retrieval systems without relevance judgments. In Proceedings of the ACM symposium on applied computing conference (pp. 811–816).

[37] Rabia Nuray, &fazli Can., (March 2005). Automatic ranking of information retrieval systems using data fusion, Information Processing and Management, 42 (2006) (pp. 595-614)

[38] M.Elena Raenda., and Umberto Straccia., Web Metasearch: Rank vs. Score Based Rank aggregation Method. In Proceeding of the ACM SIGIR conference (2003).