DESIGN OF A LEAST COST (LC) VERTICAL SEARCH ENGINE BASED ON DSHWC

THESIS

Submitted in fulfillment of the requirement of the degree of

DOCTOR OF PHILOSOPHY

to

YMCA UNIVERSITY OF SCIENCE & TECHNOLOGY

by

SUDHAKAR RANJAN

Registration No: YMCAUST/Ph13/2010

Under the Supervision of

Dr. KOMAL KUMAR BHATIA

PROFESSOR

Department of Computer Engineering

Faculty of Engineering and Technology

YMCA University of Science &Technology

Sector-6, Mathura Road, Faridabad, Haryana, India

APRIL, 2017

Dedicated

to

My Family

DECLARATION

I hereby declare that this thesis entitled "**DESIGN OF A LEAST COST (LC) VERTICAL SEARCH ENGINE BASED ON DSHWC"** by **SUDHAKAR RANJAN,** being submitted in fulfillment of requirement for the award of Degree of Doctor of Philosophy in the Department of Computer Engineering under Faculty of Engineering and Technology of YMCA University of Science and Technology, Faridabad, during the academic year May 2011 to September 2016, is a bonafide record of my original work carried out under the guidance and supervision of **Dr. KOMAL KUMAR BHATIA, PROFESSOR, DEPARTMENT OF COMPUTER ENGINEERING** and has not been presented elsewhere.

I further declare that the thesis does not contain any part of any work which has been submitted for the award of any degree either in this university or in any other university.

(SUDHAKAR RANJAN)

Registration No.YMCAUST/Ph13/2010

CERTIFICATE

This is to certify that the thesis titled "**DESIGN OF A LEAST COST (LC) VERTICAL SEARCH ENGINE BASED ON DSHWC" by SUDHAKAR RANJAN,** submitted in fulfillment of the requirements for the award of Degree of Doctor of Philosophy in Department of Computer Engineering under Faculty of Engineering & Technology of YMCA University of Science & Technology Faridabad, during the academic year May 2011 to September 2016, is a bonafide record of work carried out under my guidance and supervision.

I further declare that to the best of my knowledge, the thesis does not contain part of any work which has been submitted for the award of any degree either in this university or in any other university.

DR. KOMAL KUMAR BHATIA
Professor & Chairman
Department of Computer Engineering
Faculty of Engineering and Technology
YMCA University of Science & Technology Faridabad

Date:

ACKNOWLEDGEMENTS

First of all, I would like to thank **God**, the Almighty, for providing enough courage and blessings for completion of this thesis.

It is with deep sense of gratitude and reverence that I express my sincere thanks to my **supervisor, Department of Computer Engineering, Dr. Komal Kumar Bhatia** for their guidance, encouragement, help and useful suggestions throughout. Their untiring and painstaking efforts, methodical approach and individual help made it possible for me to complete this work in time.

I am thankful to **Dr.C.K.Nagpal, Dean (Faculty of Engineering & Technology),** YMCAUST, for constant encouragement, valuable suggestion and moral support provided by him.

I gratefully acknowledge **Dr. A.K. Sharma**, **Dr. Naresh Chauhan, Dr. Atul Mishra, Mr.Harish Kumar** a true well-wisher of mine, who always supports at every stage of completion of this thesis. I express my sincere thanks to **Dr. Manjeet Singh, Dr. Sonali Gupta, Mr.Parikshit Vasisht, Dr. Rosy Bhatia** who have always supported me and provided their valuable suggestions and for the moral support in accomplishing this task.

I am thankful to the staff members of Department of Computer Engineering at YMCAUST, for their valuable suggestions. Although it is not possible to name individual, I cannot forget my well wishers for their persistent support and cooperation.

This acknowledgement will remain incomplete if I fail to express my deep sense of obligation to my parents, wife and son (Ayush) for their consistent blessing and encouragement.

(SUDHAKAR RANJAN)

ABSTRACT

As huge amount of information is available on Web and the numbers of web sites are increasing, the quantity of pages are also increasing more rapidly.The information stored over the web is accessible through the internet. Web pages over WWW (World Wide Web) are generally classified into static and dynamic pages. The static / fixed pages fall under the category of Surface Web and the dynamic pages fall under the Hidden Web category. As the volume of contents of Hidden Web is growing exponentially, due to the popularization of the Web, customization of Web and user centric requirement increases in the Web. A lot of time is spends by the user in searching relevant web pages and the users have less time to retrieve relevant information of their choice. The complexity of data is growing and ninety five percent of the hidden web information is openly accessible without any cost to all the users but internet surfing cost are charged by the internet supplier.

The users have less time to retrieve relevant information. User spends maximum time in internet surfing to find relevant information. Searching time is directly proportional to cost of internet surfing. On the contrary, most of the time and money is spended in browsing the irrelevant information. The vertical search engine filters out the irrelevant information and supplies only the relevant information related to Hidden Web. Hence, there is a need to design and develop a Least Cost (LC) Vertical Search Engine for Hidden Web that can reduce the cost of crawling in term of minimizing response time or reduced crawling time.

The proposed architecture of a Least Cost Vertical Search Engine based on Domain Specific Hidden Web Crawler and functions helps to reduce response time/crawling time, maximizing throughput and full utilization of network bandwidth through multiple instances of hidden web crawler, load balancer technique, efficient indexing technique, ranking, query processing and next query prediction.

The proposed Least Cost Vertical Search Engine based on DSHWC works in five phases. The following phases are given below:

- Load sharing architecture of domain specific Hidden Web
- Indexing of Hidden Web contents for efficient retrieval
- Domain specific query processing
- Ranking mechanism for domain specific Hidden Web
- Predicting the next query for domain specific Hidden Web

The first phase is load sharing architecture of domain specific Hidden Web, the objective of this work is to reduce the response time/crawling time with the help of load balancer. The second phase is indexing of Hidden Web contents for efficient retrieval. The purpose of this work achieves accurate results; accelerate & searching with minimum response time. The purpose of third and fourth phases is faster retrieval of the search information. The last phase idea is to automatic knowledge extraction from the Web log and query log.

The proposed work has been implemented by using PHP and MYSQL. The results of the proposed work are found to be very promising. In order to prove this proposed work has been also compared with other research carried out in this area.

TABLE OF CONTENTS

LIST OF TABELS

LIST OF FIGURES

LIST OF ABBRIVIATIONS

WWW	World Wide Web
DSHWC	Domain Specific Hidden Web
PIW	Publicly Indexable Web
HTML	Hyper Text Markup Language
XML	Extensible Markup Language
API	Application Programming Interface
TF	Term Frequency
IDF	Inverse Document Frequency
URL	Uniform Resource Locator
LC	Least Cost
LCVSE	Least Cost Vertical Search Engine
DSIM	Domain Specific Interface Mapper
USI	Unified Search Interface
PR	Page Repository
CL	Cluster
MT	Matched Term
QT	Query Term
IMT	Inverse Matched Term
RD	Ranking Distance
RT	Relationship of Tuples

Chapter I

INTRODUCTION

1.1 GENERAL

Web search engines are essential part of the World Wide Web (WWW)[7,13]. The users use search engine to find the Web pages and containing relevant information like location searching, banking, travel, medical, auto etc. Generally, WWW or web can be divided into the Surface Web and the Hidden Web [3,4,5]. Similarly, Web pages are separated into two categories the static and dynamic pages.

The static pages fall under surface Web and dynamic pages attribute to the formation Hidden Web. Surface Web is also known as the Publicly Indexable Web (PIW) and it is searched and indexed by the conventional search engines. On the other hand the Hidden Web also known as the Invisible / Deep Web consists of very huge and versatile range of publicly accessible Web databases. The users use search engine to find Web pages. The search engine consists of three major components. These are Web crawling, indexing, and ranking as shown in Figure 1.1. Better crawling and indexing mechanism can be used to accelerate the searching process.

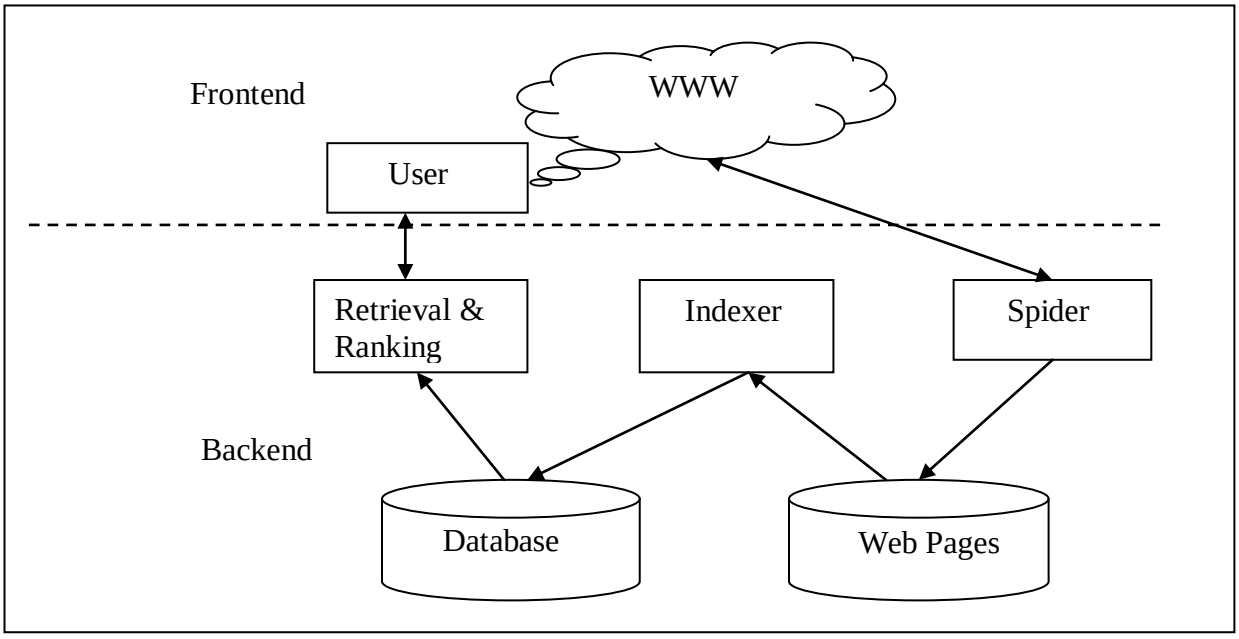


Figure 1.1: Basic Components of Search Engine

1.2 MOTIVATION

The motivation behind the work is given in this segment.

- The volume of the data increasing rapidly, the users have less time to retrieve relevant information. The user wants to use minimum time to retrieve information but they practically spend maximum time in internet surfing to find relevant information. A lot of time in term of money is wasted by internet users for searching the required information. Searching time is directly proportional to cost.

The major research objectives in the field of Least Cost Vertical Search Engine are given in the section below.

1.3. RESEARCH OBJECTIVES OF THE PROPOSED WORK

Many research articles have been published in the area to design Domain Specific Hidden Web Crawler (DSHWC). It has been found that there exists various techniques which are applied by different authors in order to reduce the cost of searching in terms of response time and utilization of bandwidth. The specific objectives of the present work are as follows:

1. **Least Cost:** The first and main objective is to design a Least Cost Vertical Search Engine based on DSHWC. Least Cost is measured in term of the following parameters:
    - Reduced crawling time / Minimizing the response time
    - Maximum utilization of network bandwidth
    - Accelerate searching through efficient Indexing
2. **Hidden Web Crawling:** The growing size of the web because of its dynamic nature; a single crawling process is inadequate. The proposed architecture helps to make a reliable, scalable and fault tolerance system. When the loads are split into the different processes, the overall accessibility is increased, performance and speed are also useful in reducing the cost in term of crawling response time.
3. **Indexing the Hidden Web Content:** The next objective is to identify and obtain the relevant information successfully and maintaining the index for

Hidden Web. With proper deployment of indexing techniques, the searching cost can be considerably reduced in terms of the decreased response time and gives effective and high precision result.

4. **Query Processing:** The vertical web provides a very large collection of independent and mixed databases, each following specific query interfaces with altered schema and query constraints, for effectively access the vertical web its required integration of these databases. The proposed method of Query processing is being proposed.
5. **Web Intelligence**: The different types of search engines already support document search support though keyword, address, and topic-based Web search. But still cannot provide high-quality intelligent services. A novel framework for web log analysis, Query log and the transaction in Hidden Web is being proposed. The data mining techniques plays a very important role for intelligent services like next query prediction, user surfing behaviour and also provides high quality intelligent services.

1.4. CONTRIBUTION

The following contributions have been made in this work to address the above challenges.

- The proposed scalable and reliable architecture and provides availability of Hidden Web crawler and also avoid pipelining of crawling process. It is helpful to reduce response time/crawling time, and in full utilization of network bandwidth.
- The load balancer distributes the hardware and software resources to the Hidden Web Crawler thereby reducing the crawling time.
- An effective and efficient Schema Instance Indexing technique has been proposed in this thesis. After implementation of indexing construction algorithm and index searching algorithms, the inter cluster communication plays very important role to obtain good processing speed.This helps in reducing search time.
- A Ranking method is to compute the ranking distance between the records. The records are organized on the basis of computing result and get relevant

record on the top. So that users spends minimum time on the searching of records and find relevant information.

- A next query prediction analysis depends upon web log analysis, Query log and the transaction in Hidden Web.
- The proposed Least Cost Vertical Search Engine based on DSHWC has been implemented using PHP technology and MYSQL. For the experimental analysis, high values of Precision, Recall and F-measure were obtained which indicates that the proposed work efficiently crawls the Hidden Web pages.

1.5 ORGANIZATION OF THESIS

This thesis consists of ten chapters, including this chapter as the introduction. Particularly, in this thesis proposed the architecture and implementation of Least Cost Vertical Search Engine based on Domain Specific Hidden Web crawler. The outline of this thesis is as follows: The first chapter is survey of Surface Web, Hidden Web, Motivation, Objective, Contribution and organization of thesis.

In chapter 2 the general framework of the web search is presented. The selected publications related to the web search covered in this thesis. This chapter contains the strength and weakness of the architecture of web search engine. Apart from this also contain the different approaches which are helpful to reduce the crawling response time and maximum utilization of network bandwidth.

In chapter 3 explores the architecture of a Least Cost Vertical Search Engine based on Domain Specific Hidden Web Crawler (DSHWC) is proposed. The overview of components has been discussed.

In chapter 4 explore the load sharing architecture of Domain Specific Hidden Web Crawler (DSHWC) is proposed. The overview of components has been discussed.

In chapter 5 presents the working of indexing techniques, comparison and proposed Index structure in details. In this chapter also present the algorithm for index construction and searching with the example of the retrieval of the data from the index.

In chapter 6 presents the working of query processing obtained the results from the proposed Schema Instance Indexing techniques and ranking mechanism and also presents the results.

In chapter 7 discusses the detailed working of ranking techniques and proposed the new technique to finding the rank of the records. It also covers the results obtained from the proposed design and also explains the results with observations.

In chapter 8 discusses the detailed working of next query prediction and association rules apply on the web log, query log for the finding the user access behavior. It also covers the results obtained from the proposed design.

In chapter 9, the various important issues in implementation of Least Cost Vertical Search Engine based on DSHWC has been discussed and in implementation used PHP and MYSQL. Various experiments over Crawling, Indexing, Query Interface, Web Log Analysis and Transaction were conducted to check the validity of DSHWC and results found are very promising. This chapter also shows the snapshots, analysis result, the different approaches to reduce the cost in term of response time and results of proposed Least Cost Vertical Search Engine based Domain-specific Hidden Web Crawler.

Finally, Chapter 10 concludes the thesis summarizes and contributions obtained through this research and provides guidelines for future work in this area.

- In Appendix A, Appendix B and Appendix C, the results search log, buy log and user log are provided.
- Finally, the bibliography includes references to publications in this area.

A literature survey of existing Web search and Hidden Web crawlers is given in next chapter.

Chapter II

WEB SEARCH: A REVIEW

2.1 INTRODUCTION

In July 1945, the theory of hypertext came to human life and the Web sites begin in 1993, before the 1993 summer, no search engine existed for the Web [7]. Generally, World Wide Web (WWW) consists of two types of Web, primarily Surface Web and secondary Hidden Web. Similarly, Web pages are separated into static and dynamic pages. The static pages fall under Surface Web / horizontal search and dynamics pages fall under Hidden Web /Deep Search. According to [6], the difference between the horizontal and vertical search shown in Figure 2.1

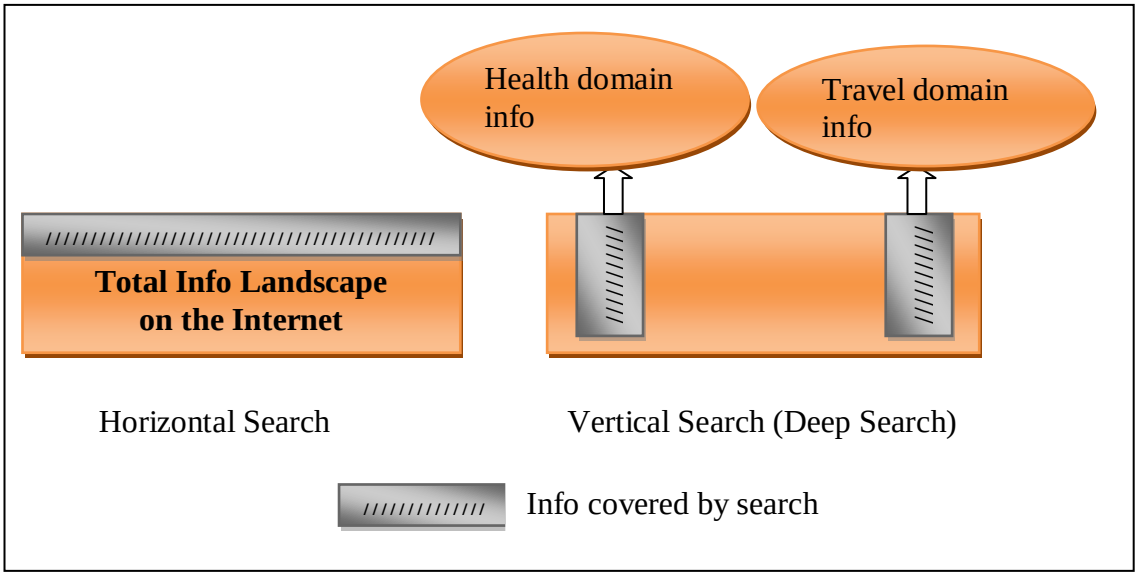


Figure 2.1: The Difference between Horizontal and Vertical Search

This source of information is available in various forms; Websites, databases, images, sound, videos and many more. Initially Web was used for research and business, but now it is used in the daily life. The number of Web pages also increasing rapidly day to day, due to popularization of the Web, customization of Web and user centric requirement increases the modification in the Web. The number of Web users increasing rapidly. The growth of the Web user and the information available in the Web creates problems in information retrieval. Web search engines are essential part of the World Wide Web. The users use search engine to find Web pages and also

finding the relevant Web pages. The World Wide Web has grown from a few thousand pages in 1993 to more than two billion pages at present. The information about the Surface Web and Hidden Web pages shown in Table 2.1

Table 2.1: Web Pages Information

S.No	Surveyor/Author	Year	Web Pages
1	Bright Planet [3,4,5]	2013, 2010, 2001	i. 19 terabytes of information belongs to surface web ii. 555 millions of unique web pages iii. 550 times larger than surface web iv. 7,500 terabytes of information belongs to hidden web v. Presently hidden web sites exists 2000,00
2	UUIC [72]	2006	i. 3,00,000 (sites) belongs to hidden web ii. 450,000 web database
3	CNNIC [44]	2008	i. 100 billion
4	Zhiguo Gong et.al [21]		i. 2,50,000 private database ii. 500 billion pages belongs to Hidden Web
5	Google &Yahoo [8]	2005	i. 11.5 million (Pages) belongs to surface web

As per the survey Hidden Web is growing exponentially on day to day basis.The complexity of data is growing and 95% Hidden Web information is publically accessible without any cost. It has also been found that approximately more than 50% of the Hidden web content resides in topic specific databases. The Figure 2.2 shown the number of total visits in google sites,yahoo sites and microsoft sites between 2009 to 2011.

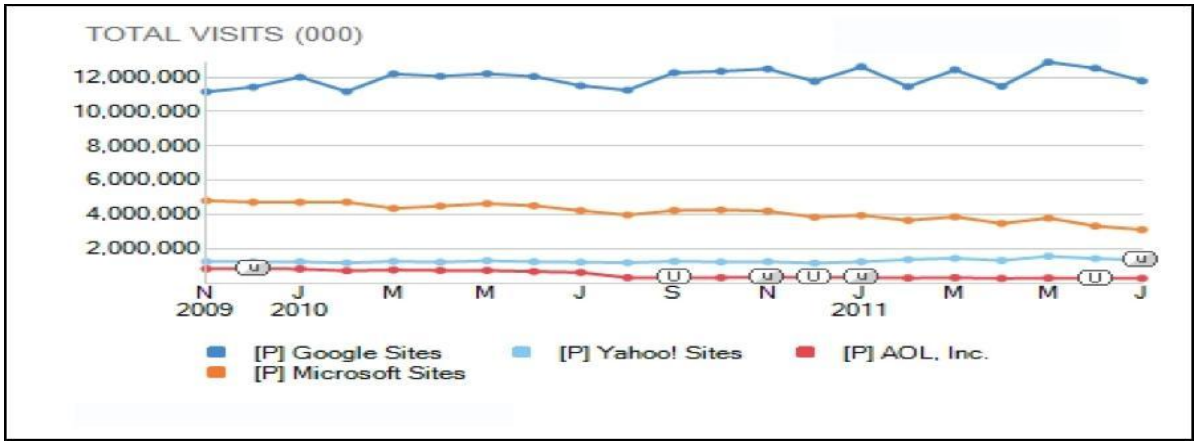


Figure 2.2: Web Sites Visitor

The total number of the unique visitors visit and total visit in billions [8] between July' 2010 to July' 2011 along with the changes in percentage shown in the Table 2.2 (Source: comScore Media Metrix. Area: United States and Europe. Measures: Total unique visitors and total visits).

Table 2.2: Total unique visitors and total visits (Europe)

Europe	**Total Unique Visitors**			**Total Visits**		
	July 2010	**July 2011**	**% Change**	**July 2010**	**July 2011**	**% Change**
Total Internet : Total Audience	353.240	370.405	5	20.065.453	21.146.307	5
Business/Finance	191.619	219.413	15	1.461.233	1.502.856	3
Credit Agricole	11.202	11.224	0	66.190	70.967	7
PayPal	18.740	21.403	14	52.147	55.558	7
Yahoo! Finance	6.561	10.361	58	34.780	61.808	78
ING Group	9.729	10.407	7	54.640	56.694	4
Lloyds Banking Group plc	8.671	9.162	6	51.940	56.092	8

The total number of the unique visitors visit and total visit in billions [8] between July' 2010 to July' 2011 along with the changes in percentage shown in the Table 2.3.

Table2.3: Total unique visitors and total visits (United States)

United States	**Total Unique Visitors**			**Total Visits**		
	July 2010	**July 2011**	**% Change**	**July 2010**	**July 2011**	**% Change**
Total Internet : Total Audience	213.584	215.054	1	15.891.604	16.767.086	6
Yahoo! Finance	41.801	39.107	-6	296.070	255.292	-14
JPMorgan Chase Property	21.455	21.686	1	120.977	114.698	-5
Business/Finance	154.773	179.914	16	1.751.621	1.824.768	4
Bank of America	24.986	25.254	1	147.756	132.031	-11
PayPal	21.116	24.114	14	70.220	73.235	4

As per the survey reports [3, 4, 5] every day Hidden Web is growing exponentially and is expected to occupy almost the entire Web share in the near future. The volume of the data is escalating rapidly and the users have less time to retrieve relevant information of their choice. The complexity of data is growing and 95% of the Hidden Web information is openly accessible without any cost to the users through internet surfing cost is charged by the internet supplier. According to report [8] which is shown in the Table 2.4 between April' 2010 to July' 2011 the various searches and the share of searches conducted on Google and Bing in the United States.

Table 2.4: Various Searches

Media/Measures	**April 2010**	**Aug 2010**	**Dec 2010**	**April 2011**	**July 2011**
Total Internet					
Searches (MM)	23.658	25.822	26.600	25.615	27.419
Share of Searches	100,0	100,0	100,0	100,0	100,0
Google Sites					
Searches (MM)	13.996	14.742	16.375	15.684	17.043
Share of Searches	59,2	57,1	61,6	61,2	62,2
Bing					

Searches (MM)	1.575	1.718	1.882	2.293	2.436
Share of Searches	6,7	6,7	7,1	9,0	8,9

It can be inferred from the Table 2.4 that in United States, Google is the leader for user preference in conducting the Web search for the relevant documents from the Web repository.

2.2 GENERAL PURPOSE SEARCH ENGINE

The general-purpose search engines such as Google, Yahoo!, and Bing etc have provided a package of services and almost cover the entire Web. Every day millions of the Web pages are crawled and indexed by the search engine. A general purpose search engine indexes everything it can find and then rank the items. The items are then presented to the user searching. It is used when searching is done for:

- Well-defined topic
- Beyond your understanding topics
- Particular site
- Web pages with millions of text
- Web sites

On the other hand, general purpose search engine have some disadvantages. As the size increases, it becomes more difficult for traditional search engines to keep an up-to-date and comprehensive search index. Moreover, the general-purpose search engines usually return thousands of pages but are not able to judge their relevance according to the topics searched by the users, and these results in degradation of accuracy of pages collection [39, 40]. Due to this the general purpose search engines are not able fulfil the information needs of the by users in specific fields. Whenever users attempt to search for some information on specific topics then the effectiveness of the general purpose engines reduces thereby increasing the time of search. This led to the birth of cleverer vertical search engines, also known as the domain specific search engine.

2.3 VERTICAL SEARCH ENGINE/ DOMAIN SPECIFIC SEARCH ENGINE

The basic purpose of the vertical search engines is to decide the relevancy at the time of the indexing itself. This is done in order to avoid adding unwanted documents to the collection [94]. There are number of proposed approaches designed to develop and implement a vertical search engine. The main approach is to identify a document is relevant or irrelevant. Figure 2.3 shows the working of the domain specific search engine / vertical search engine and general purpose search engine [45].

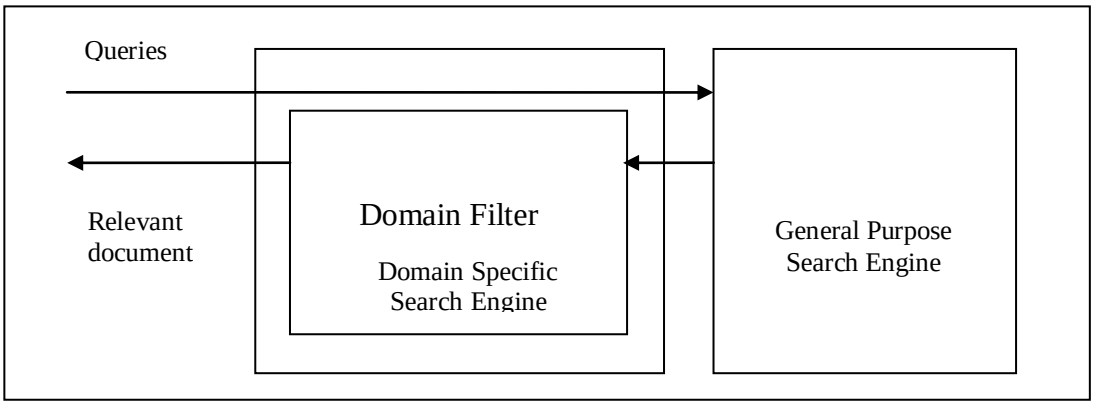


Figure 2.3: Filtering Model

One of the most popular specialized vertical search engines is *Look Smart* shown in the Figure 2.4. The few areas where the vertical search engines can be implemented efficiently are: Jobs, Travel, Health, Classifieds, Blogs, and Shopping. The Vertical search engine can be used in the following situations.

- Focusing on specific topic.
- Specific Industry, content type, geographical location, language, etc.
- When users having difficulty in locating what they want in general, meta, or concept categorizing search engines.
- When user wants the convenience of searching a variety of different content sources from one search page.

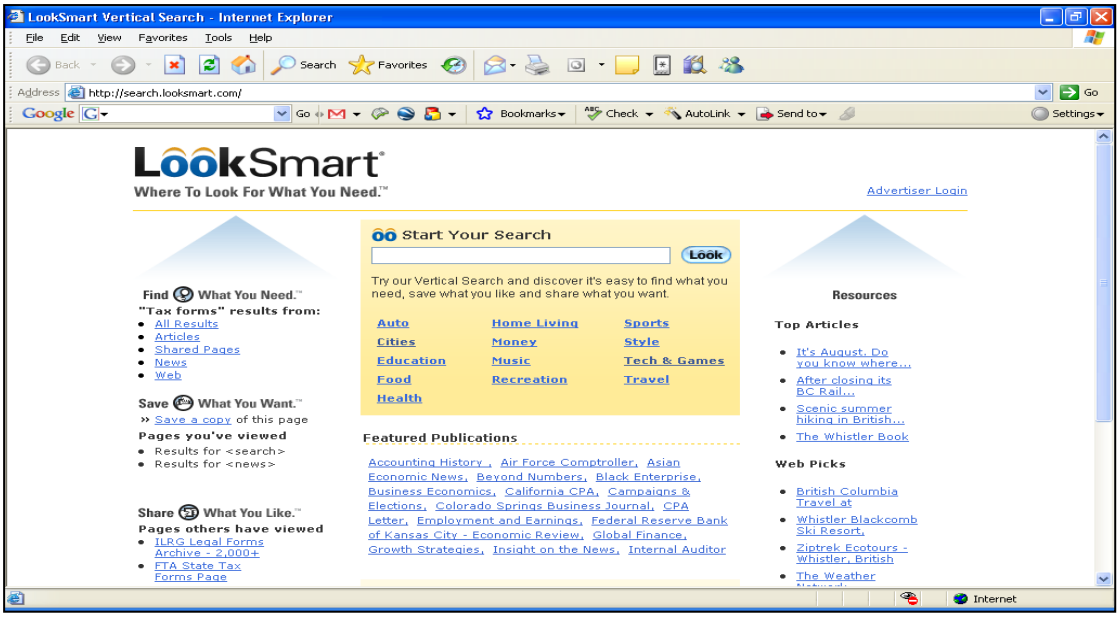


Figure 2.4: Vertical Search Engine "Look Smart"

The first component of the vertical search engine is Web crawler. It provides the way to link the request pages on the Web that are either indexed or not yet indexed. Through the search engine these Web pages are crawled and added into the index. The next component of the search engine is search interface. The working of the search interface involves three phases. Initially it accepts the user query from the Web crawler through the query interface. In second phase syntax examination is performed. Third phase involves correct spelling variations. The search interface requires database for storing the data in organised manner. A vertical search database helps in organizing the document in structured manner for fast and efficient retrieval of data. Recently, many domain-specific or language-specific search engines have been built to facilitate more efficient searching in different areas.The domain-specific search engine yields better accuracy and also save the time of search. The following are the components of vertical search technologies [6]:

- **Application Programming Interfaces**: APIs required set of routines, protocols, and tools for building software applications. Now days APIs are more popular but it is still not very common.
- **Vertical Search User Interface**. Vertical search engine can also be termed as the deep search/Domain specific search engine. Deep search is usually applied for the domains like law, games, travel, and movies, health care etc. Once the user fires the query, it is sent to the query or user interface. The user interface

is required to go through the search parameters which are dependent upon the type of application (site). For example a travel search site will require from-to, places, date, age, etc.

- **Vertical Search Algorithms**: In order to optimize the searching mechanism the user search interface employs a vertical search algorithm. The vertical search algorithm is not visible to the user but performs the functions related to the user generated query. The basic difference between the vertical search engine and general purpose search engine is its page rank algorithm. Page Rank is mainly based upon the number of hits and score of TF/IDF (term frequency/inverse document frequency).

2.4 HIDDEN WEB SEARCH

Recently, with the advancement made in the field of information technology, WWW has emerged as the leading repository consisting of huge amount of information which is readily available to the users 24 x 7 electronically. With the recurrent expansion of WWW in size, the retrieval of the relevant information has become more difficult and intricate leading to the need for more intelligent search interfaces. Hence, majority of the data becomes hidden to the users.

The Hidden Web Search has been the area of interest of many researchers. The intention to build domain-specific Hidden Web search engine is to collect only relevant data pages from the user generated queries. According to the survey report published [8] by Comscore Media Matrix , the documents stored and produced by the Hidden Web are much better in quality and also the quantity as compared to the Surface Web are much larger. Further the Hidden Web growth ratio is significantly higher than the Surface Web.

The different search types are also shown in Table 2.5categorically for both the surface and the Hidden Web. Also the improvement ratio of Hidden to Surface Web are calculated which concludes that the Hidden Web is clearly better choice as compared to the surface Web.

Table 2.5: Total "Quality" Potential, Hidden vs. Surface Web

Search Type	Total Docs (million)	Quality Docs (million)
Surface Web		
Single Site Search	160	7
Metasite Search	840	38
TOTAL SURFACE POSSIBLE	1,000	45
Hidden Web		
Mega Deep Search	110,000	14,850
TOTAL DEEP POSSIBLE	550,000	74,250
Hidden Web / Surface Web Improvement Ratio		
Single Site Search	688:1	2,063:1
Meta site Search	131:1	393:1
TOTAL POSSIBLE	655:1	2,094:1

For instance, in the case of train search process, if a user generates a query to look for any particular train. Keeping in the view to get the required information, he/she must approach the domain specific sites. Usually it is done by filling the details of the search entity in the form of search forms. As a result he/she gets the details of the trains availability and price. The very good query interfaces of IRCTC sites with various areas and airline domain from popular Web site (Yatra.com) are shown in Figure 2.5(a) and Figure 2.5(b).

Figure 2.5 (a): Query Interfaces from IRCTC

For example, www.irctc.co.in is specialized for online booking for Indian railways tickets, flights, taxi and hotels (Figure 2.5(a)).The user can search for trains that are available between the source to destination of his/her own choice.

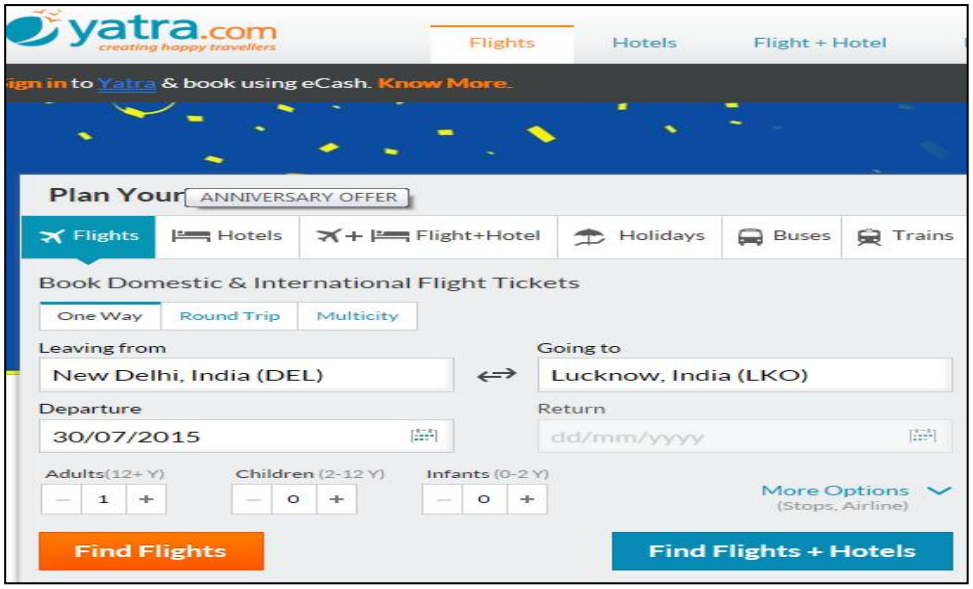


Figure 2.5 (b): Query Interfaces from Yatra.com

However, such type of search pages consists of real useful information for the user. These pages are not easily accessible to users and are universally referred to as 'Invisible'. In detail, there is an absence of the static links to such types of pages, because of which the classical search engines are unable to locate them so indexing becomes nearly improbable. The Hidden Web database contains both types of data (structured or unstructured). So, there is a need to locate such type of pages for the complete information retrieval.

2.4.1 Hidden Coverage Area

In year 2004, a study carried out by the University of California 'Berkeley' around 3,00,000 Hidden Web sites and as per UUIC survey and research publications [72] the number of Hidden Websites amount to 3,00,000 and Web databases approximate to around 4,50,000. However, there were only around 14,000 Hidden Web sites reported in year 2006 in the Russia. As per the global internet statistical information released by CNNIC [44], the number of Web pages is estimated to exceed 100 billion [3,4,5,21]. As per the survey conducted by Bright Planet Company in 2013, 2010 and 2001[3,4,5] the internet is extensively bigger with an estimated 555 million

domains[3,4,5,15], each comprising of thousands of unique pages and is classified into many areas. The division among different areas is shown in Figure 2.6. It may be noted that the coverage percentage for “Humanities” is the highest and for “Agriculture” is lowest (The highest coverage is 13.5% for humanities and the lowest coverage 2.7% for agriculture). According to the literature review [4] the Hidden Web is classified in total of 18 categories as listed in the Table 2.6. The highest coverage percentage of Hidden Web is in the field of humanities (13.5%) and the lowest in the domain of agriculture (2.7%).

Table 2.6: Hidden Coverage Area

S.No	Subjects	Coverage (%)
1	Humanities	13.5%
2	News, Media	12.2%
3	Computing/Web	6.9%
4	Arts	6.6%
5	Business	5.9%
6	Agriculture	2.7%
7	Education	4.3%
8	Employment	4.1%
9	Engineering	3.1%
10	Government	3.9%
11	Health	5.5%
12	Shopping	3.2%
13	Law/Politics	3.9%
14	Lifestyles	4.0%
15	Travel	3.4%
16	People, Companies	4.9%
17	Recreation, Sports	3.5%
18	References	4.5%
19	Science, Math	4.0%

As per Table 2.7, there are sixty largest well known Hidden/Deep Web sites, these Web sites consists of huge amount of data reaching up to approximately 750 TB. This is roughly 40 % size of the surface Web. These Web sites cover a broad array of domains ranging from engineering, humanities, agriculture, business, company and more. Also, the estimate [3,4] indicates that there are approximately 85 billion numbers of records or documents available in this group. Moreover, nearly two-third

of these sites is public. Table 2.7, shown that the amount of data in Hidden Web is very large as compared to the Surface Web and lists the name of the sites along with their URLs, type of size they occupy on the Web. According to Table 2.7, there are 17000 deep Websites covering all the subjects used in the study. The table indicates that there is a uniform distribution across all domains and content in all the categories are represented significantly.

Table 2.7: Hidden Web Sites Name, Size and URL

Name	**Type**	**URL**	**Web Size (GBs)**
National Climatic Data Center (NOAA)	Public	http://www.ncdc.noaa.gov/ol/satellite/satelliteresources.html	366,000
NASA EOSDIS	Public	http://harp.gsfc.nasa.gov/~imswww/pub/imswelcome/plain.html	219,600
National Oceanographic	Public/Fee	http://www.nodc.noaa.gov/, http://www.ngdc.noaa.gov/	32,940
----------	----------	----------	-------
Subtotal Public and Mixed Sources			673,035
DBT Online	Fee	http://www.dbtonline.com	30,500
---------	----------	----------	-------
Subtotal Fee-based Sources	----------	----------	75,469
Total			7,48,504

Figure 2.6 show that more than 50% of all Deep/Hidden Web sites attribute relevant / current databases. The topical databases, internal site pages and stored publications cover roughly 80% of all Deep / Hidden Web sites. Rest of the nine categories combine to form remaining 20% of Web.

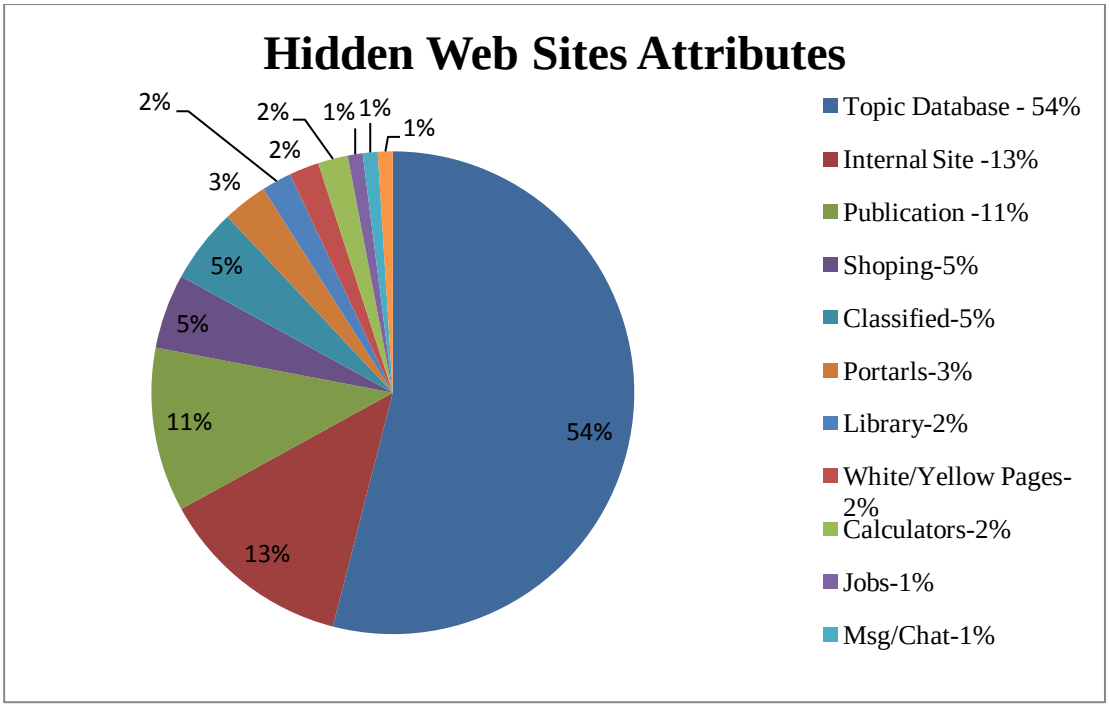


Figure 2.6: Hidden Web Sites Attributes

2.4.2 Higher Quality

The quality of the Hidden Web documents can be classified into subjective values rated as high quality, low quality and no quality shown in Figure 2.7. These classifications are usually dependent on the downloaded document relevance which can be further be measured in terms of computational linguistic score in the case of Hidden Web as well as the surface Web.

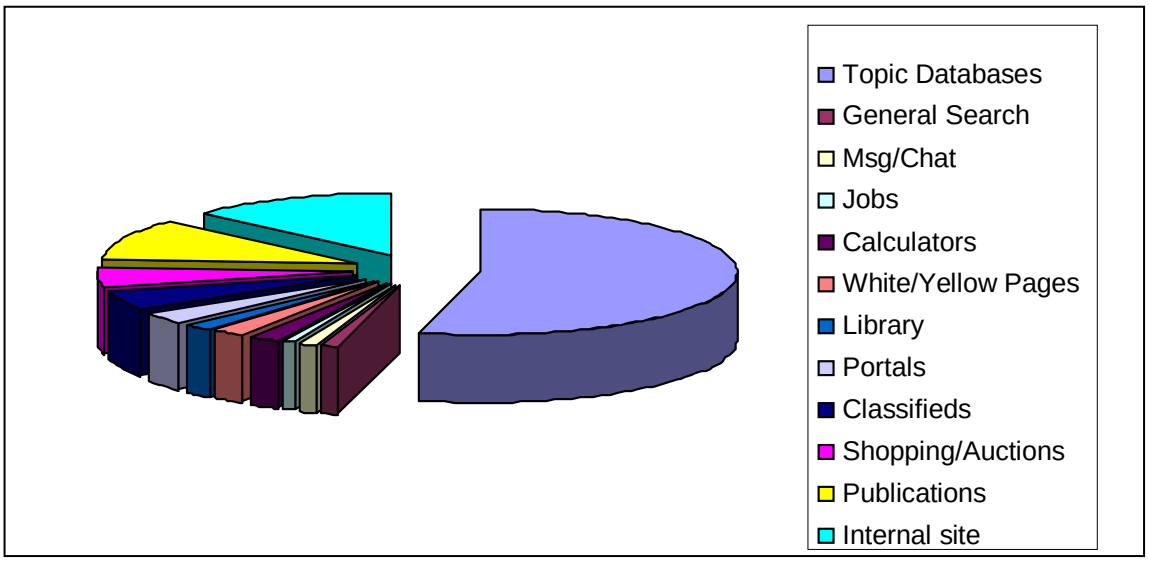


Figure 2.7: Hidden Web Document Classification

As the same criterion is applied in case of both of Hidden Web and Surface Web, it becomes very difficult to distinguish between the quality of Hidden Web and the Surface Web. By applying, filtering techniques and tests in addition to the computational linguistic scoring, quality results have been produced by Bright Planet [4]. Table 2.8 shows the output of queries generated across different domains, in teem of yield.

Table 2.8: Output Query Result

Surface Web				**Hidden Web/Deep Web**		
Query	**Total**	**"Quality"**	**Yield**	**Total**	**"Quality"**	**Yield**
Science	700	30	4.3%	700	80	11.4%
Finance	350	18	5.1%	600	75	12.5%
Medicine	500	23	4.6%	400	50	12.5%
Agriculture	400	20	5.0%	300	42	14.0%
Law	260	12	4.6%	320	38	11.9%
TOTAL	**2,210**	**103**	**4.7%**	**2,320**	**285**	**12.3%**

According to the Table 2.8, it is inferred that the amount of quality data contained inside the Hidden Web is almost three times of that contained within the Surface Web. It can be also seen that about 10% more documents are returned from the Hidden Web as compared to the Surface Web. It can be reiterated that the Hidden Web is considered as a huge potential source for providing the high quality results. Also it is noted that with increase in the number of Web sites searched, there is a considerable rise in the quality of the documents yielded.

2.4.3 Hidden Content Growth

In [3, 4,5, 21] it is stated that at present, the World Wide Web has grown from a few thousand pages in 1993 to more than two billion pages. The survey conducted by Bright Planet company in 2013,2010 and 2001 the internet is extensively bigger with an estimated 555 million domains, each containing thousands or millions of unique Web pages and other observations[4] are given below:

- The Hidden Web size is 550 times larger than the surface Web.

- The Hidden Web contains 7,500 terabytes of information and surface Web contain 19 terabytes of information.
- The Hidden Web contains around 550 billion individual documents and Surface Web contains one billion individual documents.
- Presently Hidden Web sites that exist are more than 200,000.
- Total quality content of the Hidden Web is 1,000 to 2,000 times greater than that of the surface Web.
- Sixty of the largest Hidden Web sites collectively contain about 750 terabytes of information sufficient by themselves to exceed the size of the Surface Web by forty times.
- An average, Hidden Web sites receive fifty percent greater monthly traffic than Surface Web sites and are more highly linked than the Surface Web sites. However, the typical Hidden Web sites are not known to the searchers.
- The Hidden Web sites tend to be narrower with deeper content than the conventional Surface Web sites.

As per the survey Hidden Web is growing exponentially on day to day basis.The complexity of data is growing and 95% Hidden Web information is publically accessible without any cost. It has been found that approximately more than fifty percent of the Hidden Web content resides in topic specific databases.

2.4.4 Hidden Web Crawling Techniques

As per the suggestions made by the various authors in the research articles the operation of the Hidden Web crawlers has been described in four to five phases which are discussed in Table 2.9.

Table 2.9: Hidden Web Crawler Techniques

S.No	Paper Title	Technique used
1	A Framework of Deep Web Crawler [14] (Xiang Peisu et.al)	i. Form Analysis ii. Value assignment and submission iii. Response analysis iv. Response Navigation

2	AKSHR: A Novel Framework for a Domain-specific Hidden Web Crawler [15] (Komal Kumar Bhatia et.al)	i. ii. iii. iv.	Search Interface Crawling Domain-specific Interface Mapping Automatic Form Filling Response Page Analysis
3	Finding the WDB's Query Interface in Deep Web Automatically [16] (Peiguang Lin et.al)	i. ii. iii.	Extract Features of Query Forms Extracting Query Interface Features of domain query interfaces Identifying and Classifying Query Interfaces
4	Hidden Web Database Exploration [21] (Zhiguo Gong et.al)	i. ii. iii. iv. v.	Hidden Web Pages Parse Web Pages Generate the Knowledge Base for a specific domain Analyze the Response Pages The Relevance of a hidden database
5	A Crawler for Local Search[17] (Pedro Huitema et.al)	i. ii.	Determine Relevancy Crawling Behavior
6	Domain-specific Web Service Discovery with Service Class Descriptions [110] (Daniel Rocco1 et.al)	i. ii. iii. iv. v. vi.	Search Form Filter Form Interface Analyzer Module Selection Query Generator Query Selection & Probing Response Matching
7	A Vertical Search Engine – Based On Domain Classifier [18] (Rajashree Shettar et.al)	i. ii. iii. iv. v. vi. vii.	Web Crawler HTML Parser Filter Domain Classifier Page Ranker URL DB Search
8	A Distributed Approach To	i.	Analyzer

	Crawl Domain Specific Hidden Web [12] (Lovesh Kumar Desai et.al)	ii. Parse iii. Composer iv. Result Analyzer
9	Crawling the Hidden Web [19] (Sriram Raghavan et.al)	i. Classify Dynamic Web Content ii. Modeling Forms and Form Submissions iii. HiWE: Hidden Web Exposer iv. Design Issues and Techniques
10	Exploiting Ontology for Retrieving Data Behind Searchable Web Forms [20] (A.IEL-desoky et.al)	i. Form Processing ii. Form Analysis iii. Page Classification iv. Matching with ontology v. Form filling & automatic query generation vi. Response Analysis
11	How Search Engines Work and a Web Crawler Application [13] (Monica Peshave et.al)	i. Working of search engine ii. Working of Web crawler iii. Meta search engine iv. Indexing

2.4.4.1 AKSHR: A Domain Specific Hidden Web Crawler

AKSHR [15] is a domain specific Hidden Web crawler which provides fully automatic techniques to download the search interfaces and matches them by using the DSIM framework as shown in Figure 2.8. A brief description of each phase is given below:

- In the first phase, a novel algorithm to collect and index Web pages that will act as entry points to the crawler is proposed. The Search Interface Crawler crawls the search interfaces and stores them in Search Interface Repository for further use.
- The second phase introduces a novel technique to automatically identify (detect) semantic mappings between the attributes of search interface and the labels used in Domain-specific Data Repository to fill the search

interfaces.

- In the third phase, a Domain-specific Data Repository containing labels and their corresponding values has been employed that helps in automatically filling the interfaces one by one.

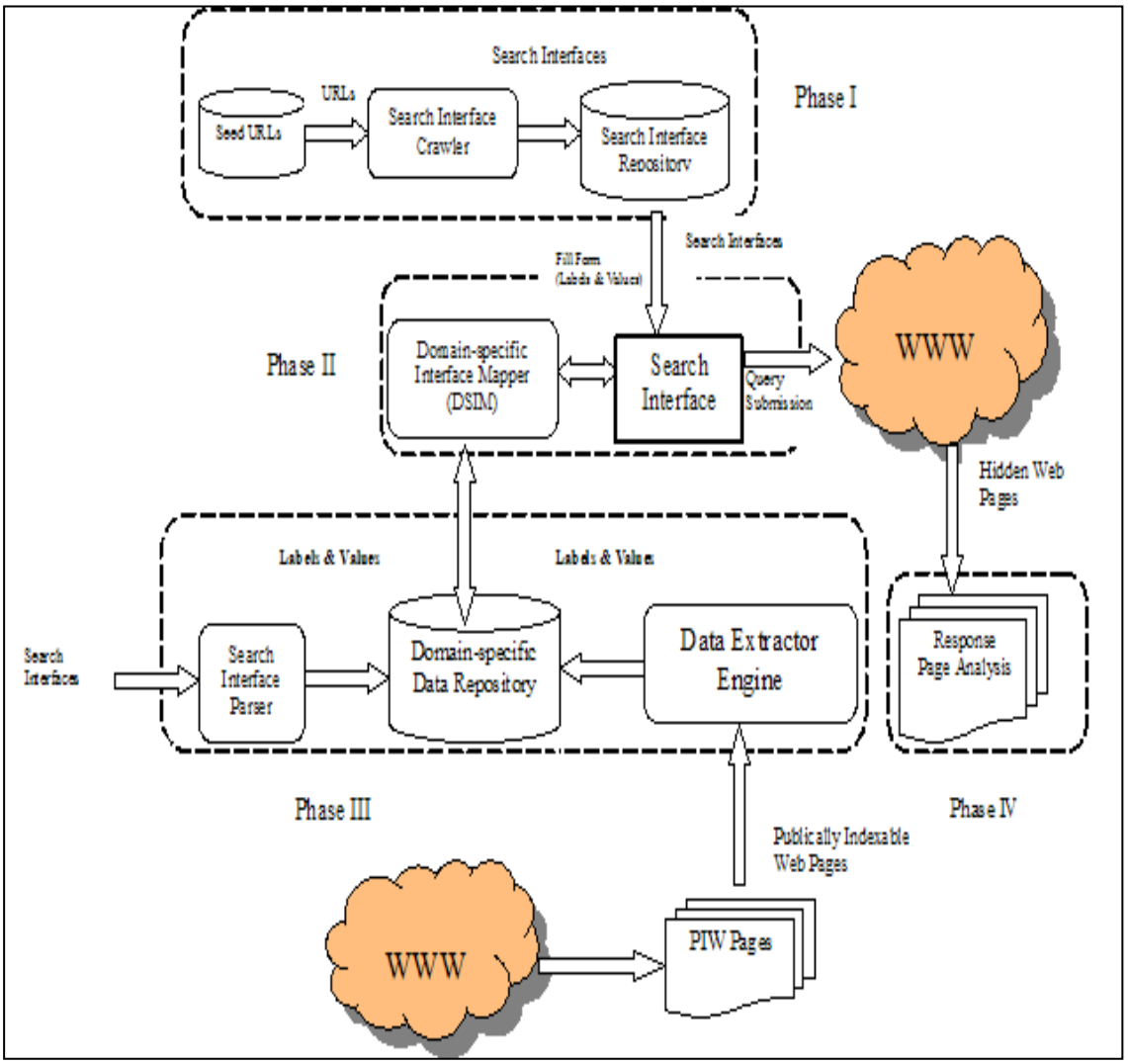


Figure 2.8: Architecture of a Domain Specific Hidden Web Crawler (AKSHR)[15]

- The last phase analyzed the response pages with a view to filter the erroneous pages.

2.4.4.2 Research and Design of the Crawler System in a Vertical Search Engine

The benefits and key points of the multithreading crawling system in vertical search engine [11], concepts given below:

- Retrieving huge data
- Synchronized accessing of shared resources
- Processing efficiently
- Waiting time in accessing query reduced

The work flow of multithreading crawling system in vertical search engine shown in Figure 2.9, the multithreading crawling system contains five components.

- The Crawler System
- Web Process Database
- The Index Module
- Indexing Database
- The Web User Interface

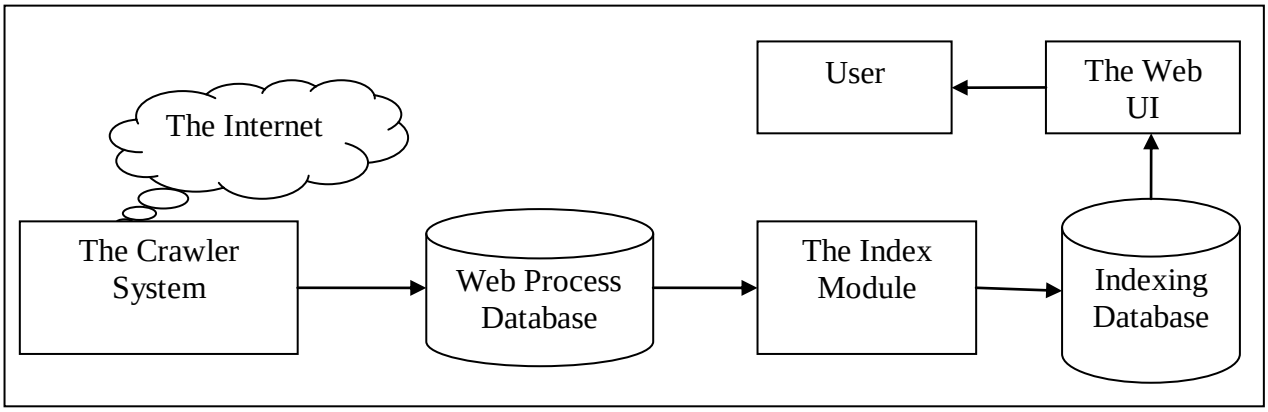


Figure 2.9: Workflow of Vertical Search Engine

2.4.4.3 Architecture and Implementation of an Object Level Vertical Search

A brief description about the system architecture of an object level vertical search engine [23] shown in Figure 2.10, the importance of object level vertical search engine given below:

- Crawled data classified into different categories and extractor pull out metadata.
- Object level ranking and mining techniques applied to construct search more precise and intelligent.
- Every extracted data need to be mapped to a valid object and stored in repository.

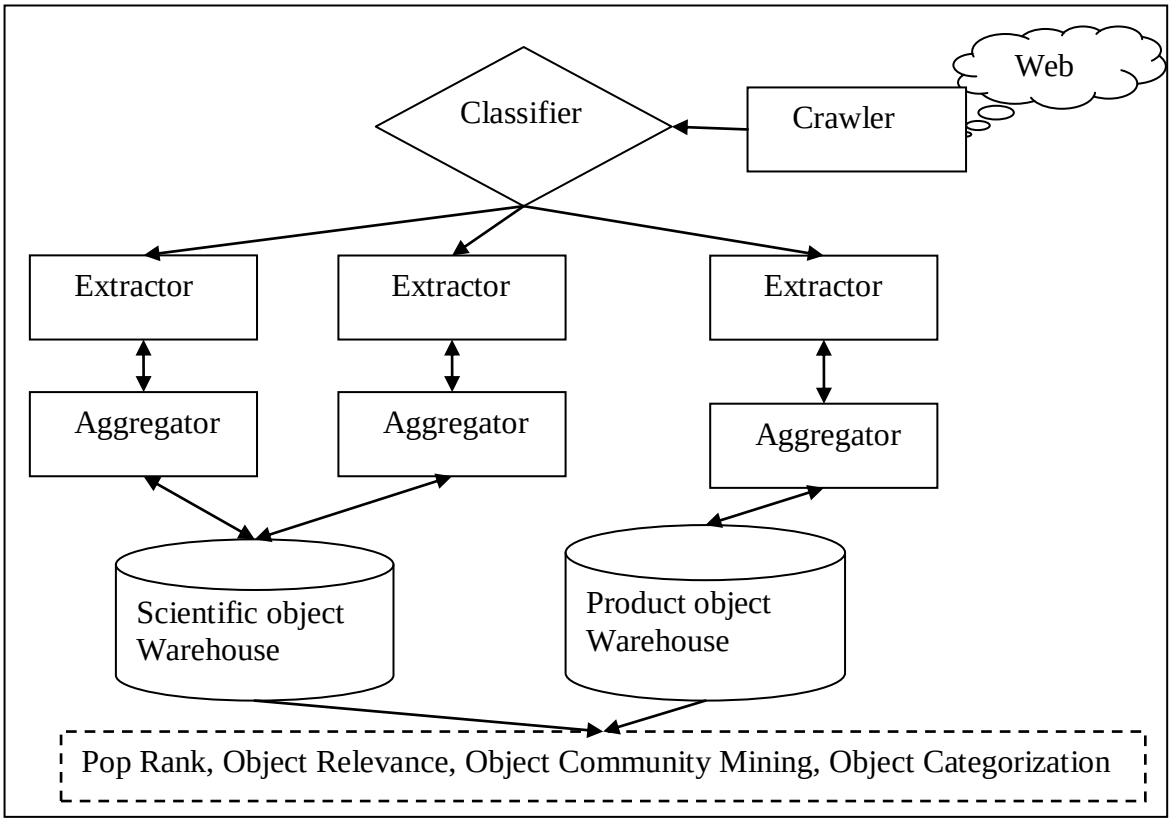


Figure 2.10: Architecture of Object Level Vertical Search

2.5 LEAST COST TECHNIQUES

Many research articles have been published to design least cost search engine. It has been found that there exist various techniques which are applied by different authors in order to reduce the cost of searching. Following are the most widely employed techniques:

- Load Balancer
- Efficient Indexing Technique
- Ranking
- Query Processing
- Next Query Prediction

2.5.1 Load Balancer

The load balancer [1,71] plays the vital role in the reduction of the search cost. It can be defined as the technique employed at the various instances where the optimization of the available resources is achieved. Other important features provided by the load balancer are specified as follows:

- Recourses scaled up on demand
- Distributes workload
- Maximum utilization of resources
- Minimizing the response time
- Maximizing throughput
- High Availability

Further the load balancer can be classified in terms of the following categories mentioned in the Table 2.10 along with the merits and demerits.

Table 2.10: Load Balancer Categories: Merits and Demerits

S.No	**Load Balancing Categories**	**Sub-Categories**	**Merits /Demerits**	**Method**
1	Static	No	Load assigned before application runs	Round Robin Randomized Partitioning
2	Dynamic	Centralized	Load assigned as application run	Master-Slave
		Decentralized	Tasks reassigned among slaves	Receiver Initiated Sender Initiated
3	Semi-Dynamic	No	Application periodically suspended and load balanced	Partitioning

According to the literature survey [1] there are basically two types of load balancers categorised as the software balancer and the hardware balancer. There are four products of software balancers available out of which Nginx occupy the major share of about 50%. Some of the merits of using hardware balancers are that they provide high quality network services. According to Cisco Systems nearly 45000 customers use Cisco system catalyst 6500. Similarly there are other three types of hardware load balancers are available as shown in the Table 2.11.

Table 2.11: Load Balancer Type

S.No.	Load Balancer	Type
1	Cisco System Catalyst 6500	Hardware
2	Barracuda Load Balancer	Hardware
3	F5 Network BIG-IP LTM	Hardware
4	Coyote Point Equalizer	Hardware
5	Nginx	Software
6	HA Proxy	Software
7	Pround	Software
8	Varish	Software

The Table 2.12 depicts the working principle of routing methods.

Table 2.12: Load Balancer Routing Method and Working Principle

S.No.	Routing Method	Working Principle
1	Round Robin	Select one by one
2	Weighted Round Robin	Server assigned some weight
3	Priority	The incoming traffic is routed to high priority server
4	Overflow	Same as priority method but in overflow condition, routed at low priority server.

2.5.2 Indexing Techniques

As per the literature review [4], the Intelligent Indexing has successfully replaced the manual indexing. It is governed electronically and is much faster as compared to the traditional indexing. The incomplete indexing of surface Web sites is shown in Table 2.13.

Table 2.13: Incomplete Indexing of Surface Web Sites

Engine	OPD Pages	Yield
Open Directory (OPD)	248,706	-
AltaVista	17,833	7.2%
Fast	12,199	4.9%
Northern Light	11,120	4.5%

With proper application of indexing techniques and implementation of indexing algorithms, the searching cost can be considerably reduced in terms of the decreased response time. As suggested by literature survey in [26,27,28,29,30,31,32] there are several techniques applied for indexing mechanism. Some of the techniques of indexing are discussed below:

- **Inverted index:** It is defined as the structure that holds the information about a specific term in much elaborated form. The index contains number of occurrences of a particular term or the total number of documents related to that term. It accelerates the ad hoc searches. It is unable to search the key values but is helpful in database searching. Localization of required data without being actually going through all the data is provided by indexing. Enhancement of efficiency and improved performance of indexing helps in query processing.
- **Bitmap index:** This technique involves a representation of index by using a series of bits and is widely used in designing of the data warehouses.
- **Value list Indexing:** The encoded scheme of bitmap index is referred to as Value list Indexing. It consists of two parts: Balance Tree structure and mapping schema.
- **Direct/Forward Index:** It is also known as the inverse form of the inverted index. It holds the information about the term such as term id and frequency for every document inside the collection. Similarly by applying compression the storage space can be saved as in the case of inverted indexing. It lists all the terms contained within the document.
- **Projection Index:** It simply copies the table and maintains the table in the same original order.

As per the literature review [28] the intelligent Indexing process broadly is fragmented into four steps classified as pre-treatment, analyzing text, index sorting and searching.

- **Step1: Pretreatment:** This is the process of extracting the text document from the non text document as only trivial text document is needed for indexing.

- **Step 2: Analyzing Text:** In this, the text is converted into indexed representation. This requires extracting words, discarding punctuation, removing common words, stop-word removal, and tokenisation.
- **Step 3: Index sorting:** In this step, the data is stored in database utilising the inverted data structure. Pretreatment and text has already been performed and thereafter sorting process is applied in that data.
- **Step 4: Searching:** It plays the important role in the identification of the index words through enhanced search algorithm. As per the literature [28,29] for better searching and efficient results, intelligent search engines and optimization algorithms of searching are required.

Index structure for efficient retrieval of information for general / invisible Web for many techniques has been proposed in Table 2.14 provides advantages and disadvantages between them.

Table 2.14: Indexing Techniques: Advantage and Disadvantage

S.No	Index Techniques	Advantages	Disadvantages
1.	Inverted Index	i. Accelerated ad hoc search ii. Less expensive	i. Store more detail data ii. More space
2.	Bitmap Index	i. Less Space ii. Easy updating iii. Grouping	i. Less efficient for large data ii. Expensive
3.	Value list Index	i. Very efficient for query ii. Independent space requirement iii. Less expensive	i. No grouping ii. No ad hoc query support
4.	Cluster Index	i. Improve the efficiency ii. Increase the speed of query Grouping	i. No ad hoc query support ii. Expensive iii. More space
5.	Projection Index	i. Reduce query cost Fast access	i. Not efficient for query
6.	Direct/Forward Index	i. Less storage space	i. No ad hoc query support
7.	Document Index	i. Keep all information	ii. No ad hoc query support

To make advance or intelligent indexing system requires index to be created on a single/multiple attributes in order to accelerate the indexing performance

2.5.3 Query Processor

According to [53,54,55,56,57], user interacts with the query interface. It allows the user to access information from the Web database. Structured Web databases can be inquired through the query forms, Web service interfaces, and query interfaces; end users are able to express their information needs by imposing selection conditions on certain attributes of interest. After this the query reaches the query processor. It converts the query into sub query(s) by implementing precise plan which expresses these sub-queries in the form of low level languages. The query processing is segregated into two discrete levels. The first level is syntax checking where the objects in the query syntax are compared with views, relation and columns listed in the system's relation. This level employs methods like tokenisation and stop-word removal. The purpose of tokenization is divide the documents into the individual words and also remove the lower case letters and punctuations. The stop-word removal as the name suggests omits the common words, such as 'and' and 'the', these are very frequent and their presence or absence does not influence the result. The second level is known as query modification level. In stemming each word converted to its root form by removing suffix and prefix. For example 'write', 'writing', 'wrote' may be changed to "write".

2.5.3.1 Term Matcher

The term matcher constitutes several models like Boolean model, Word list, Data type matching, Vector space model and statistical language model [15,62, 63]. The Boolean model works on the principle of exact matching in which the documents are matched according to the user's query wherein words are logically combined by using Boolean operators like AND, OR, and NOT. This method is the earliest and precise model. For example, applying the Boolean operator AND on any logical statements x and y will provide the output that satisfies both the attributes of statements x and y. However, in case of the Boolean OR either of these statements must be satisfied i.e. either x or y. Vector space models consists of transforming the textual data into numeric vectors and matrices. Once this is done, matrix analysis techniques are

employed to discover key features and connections in the document collection. Statistical language models attempt to estimate the probability that the user will find a relevant document. Retrieved documents are ranked by their odds of relevance (the ratio of the probability that the document is relevant to the query divided by the probability that the document is not relevant to the query).

2.5.4 Ranking Mechanism

The success of the search engine depends upon its ability to rank Web pages given in the result. The ranking algorithm helps to find out the frequency of key words on the Web page. Apart from this, linking of the Web pages is provided through the algorithms. The Web search page relevancy depends upon the search terms which have appeared in the HTML title tag. The other most important factor of search engine attaining relevancy is frequency. The rank is calculated for each individual term stored along with the URL in the database. Ranking is achieved through a number of ranking methods like Page-Rank, Score, Hits, etc., approach of each of these is different in computing the rank score. In search engine, Rank calculator is one of the most significant features. The ranking module computes the rank by sorting the relevant pages applying rules using combination of two scores, the content score and the popularity score. This yields the output in the form of the ordered record comprising of Web pages in such a fashion that the pages on the top possess the highest rank. The *popularity score* is computed on the basis of the analysis carried out in the Web's hyperlink structure. The set of relevant pages resulting from the query module is then presented to the user in order of their overall scores.

2.5.4.1 Term Frequency-Inverse Document Frequency

TF-IDF is widely used technique used for the computation of ranks allocated to Web search pages. It is based on the principle that occurrence of the term is directly dependent on its frequency also referred to as Term Frequency. The Inverse document frequency is the inverse of the total number of the documents. Documents that match a user given query may be ranked based on how well they match the query. The relative frequency or TF-IDF score can be calculated as per [18] the relation given below.

$$\mathrm{T}FIDF = TF(term\ frequency) * IDF(inverse\ document\ frequency) --eq.2.1$$

$$TF = \frac{(no.of\ times\ lexicon\ word\ in\ the\ page\ occurs)}{(total\ no.of\ words\ found\ in\ that\ pages)} --\mathrm{eq.}2.2$$

$$IDF = \frac{toal\ no.of\ web\ pages}{no.of\ web\ pages\ in\ which\ the\ lexicon\ word\ occur} -- eq.2.3$$

2.5.4.2 Probability Ranking Principle

Using a probability Ranking Principle, the document is ranked on the basis of their approximate probability which defines the importance of documents with respect to the information needed. As per the paper [40], *Page A has pages T1...Tn which point to it (i.e., are citations). The parameter d is a damping factor which can be set between 0 and 1. Usually set d to 0.85. Also C(A) is defined as the number of links going out of page A. The Page Rank of a page A is given in* equation (2.4).

$$PR(A) = (1-d) + d\left(\frac{PR(T1)}{C(T1)} + -- + \frac{PR(Tn)}{C(Tn)}\right) -- eq.2.4$$

Page Rank or *PR(A)* is calculated using a simple iterative algorithm, and points to the principal eigenvector of the normalized link matrix of the Web. Other advantage of this technique is that the Page Rank for about 26 million Web pages can be computed in a few hours on a medium size workstation [40].

2.5.4.3 Page Link Rank Method

In this method the page P, contains the incoming links referred as the “in links” [43,44] shown in Figure 2.11. The score in this method is calculated by applying the Web content and link analysis techniques. One of the widely quoted demerits of this method is that it is expensive when used for computation of the rank.

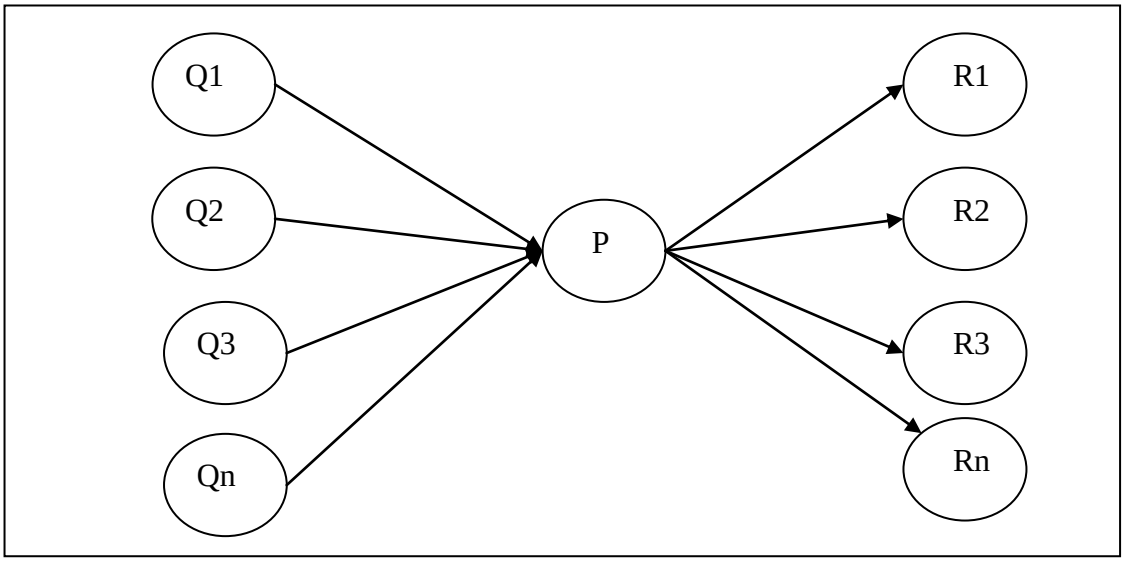


Figure 2.11: Link-based Method

Presently it is in effective use in case of general purpose search engine. However, in case of the Hidden Web it is costly.

2.5.4.4 Entity Level Ranking Mechanism

The entity level ranking [38] model is classified into the three layers shown in Figure 2.12. The bottom layer is defined as extraction layer, the middle layer is known as the local score layer and the top layer are called the global aggregation layer.

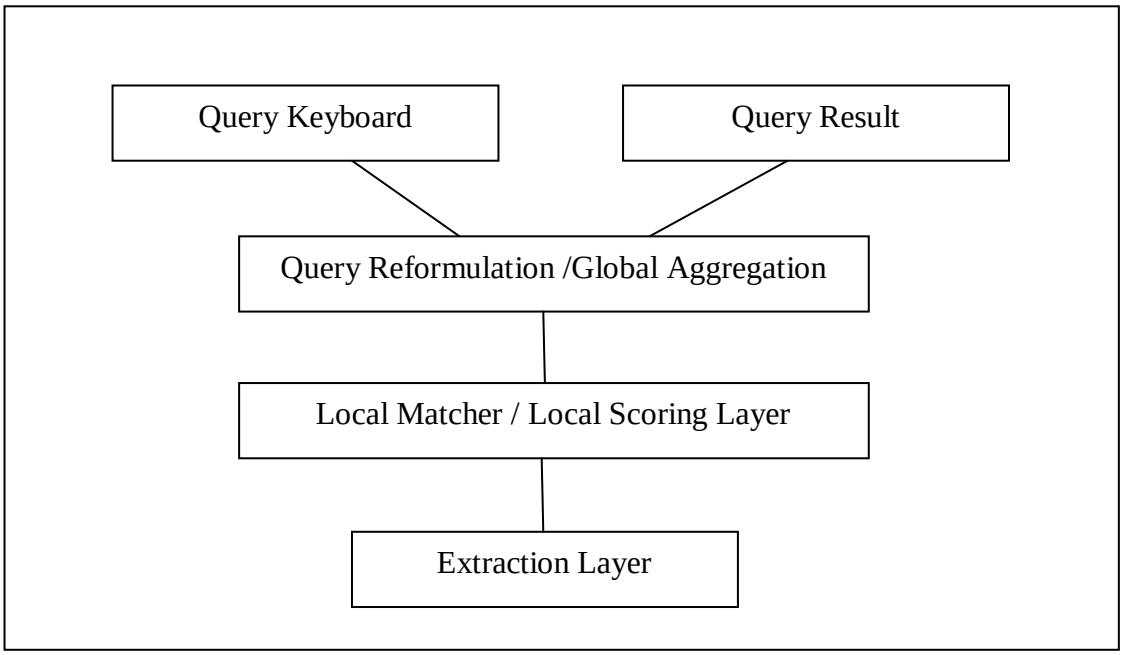


Figure 2.12: Entity Level Ranking

The local scoring layer is widely used for Web pages and attributes like title, year etc. If the entity appears exists in multiple Web resources, then the frequency of the entity becomes the main factor to calculate score. The demerit of the global aggregation is results in duplication or repetition of the entity and leads to the inaccurate frequency.

2.5.4.5 Weight Rank

In the traditional search engine, the ranking system is maintained using more information related to the Web documents. Almost every Web document include following features like position, font, capitalization information, anchor text and the Page Rank. This is known as hit. The ranking mechanism becomes difficult because of the need to combine all this information. Google [40] considers a hit to be at any of the location i.e. title, anchor, URL, plain text each possessing its own type-weight. As the number of hits rises, count weights increase linearly. At the end the IR score combined with Page Rank is used to allocate the final rank to the document.

2.5.5 Next Query Prediction

As per [98,99,100,101,102,103,104,105,106,107] it has been stated that according to the user preferences, the user is allowed to access Web pages. In this work, access log data is randomly generated for analytical purposes. After the file is identified, Logging is performed. The access log and session files are maintained. The Association Rule (Apriori Algorithm) is used to extract and predict frequent path and user access behaviour according to the rules. The Apriori Algorithm, a well-known approach proposed by Agrawal & Srikant et.al (1994) is used to find frequent item sets, which are defined as the group of items occurring frequently and simultaneously in many transactions. The problem of mining association rules is divided into two parts: all frequent item sets are found i.e. finding all combinations of items that have transaction support above a support threshold, and generating the association rules from these frequent item sets.

Once the transactions have been identified, the association rules are applied to the relationship among this data set. The Apriori algorithm has been used to predict the next query. As the result of surveys conducted [1, 2, 9, 10, 11, 34] there are different

methods that help in designing a Least Cost (LC) Vertical Search Engine based on DSHWC. Least Cost is measured in term of the following parameters:

1. Reduced crawling time / Minimizing the response time
2. Maximum utilization of network bandwidth
3. Efficient Indexing for better searching

The volume of the data is increasing rapidly and the users have less time to retrieve relevant information. Majority of the time is spent in internet surfing to retrieve relevant information. Running time is directly proportional to the cost. On the contrary, most of the time and money is spent in browsing the irrelevant information that is reaching the user along with the genuine information required. The vertical search engine filters out the irrelevant information and provides only the relevant information. DSHWC has higher precision and accuracy as compared to general search. The proposed architecture thus provides the key that lay emphasis on saving the search time of the user and thus attain higher cost effectiveness.

The next chapter provides the overview of proposed architecture of a Least Cost Vertical Search Engine based on Domain Specific Hidden Web Crawler (DSHWC) for Hidden Web.

Chapter III

PROPOSED ARCHITECTURE OF LEAST COST (LC) VERTICAL SEARCH ENGINE BASED ON DSHWC

3.1 INTRODUCTION

As huge amount of information is available on World Wide Web (WWW) and the numbers of Web sites are increasing, the quantity of pages is also increasing more rapidly.The information stored over the Web is accessible through the internet. Web pages over WWW are generally classified into static and dynamic pages. The static / fixed pages fall under category of surface web and dynamic pages fall under Hidden Web category. As the volume of Hidden Web is growing exponentially, a lot of time is spent by the users in searching relevant Web pages. For accessing, searching and retrieval of the Web information a search engine is generally required. The conventional search engines classify and index only static pages.The dynamic pages are not indexed by the conventional search engines. The general purpose search engine does not work effectively for finding relevant Web pages over the Hidden Web.

In this work, this drawback has been removed by the proposing vertical search engine which operates on the principle of finding the topic relevant pages there by leading to the better quality of Web search for Hidden Web. The “Hidden Web” or “Invisible Web” contents are currently not a part of conventional search engine. The searching of the Hidden Web is very difficult due to the two basic reasons; the first issue is size of the content stored in online database and secondly, it requires access to the database through restricted search interfaces so as to extract relevant content. This increases the cost of accessing, searching and retrieval is generally the cost of accessing the relevant data is measured in term of the following parameters:

1. Crawling Time / The Response Time
2. Utilization of Network Bandwidth
3. Efficient Indexing for better Searching

The user wants to spend minimum time to retrieve information but they practically spend maximum time in internet surfing to find relevant information. Searching time is directly proportional to cost of crawling and searching in index (crawled repository). Hence, there is a need to design and develop a Least Cost Vertical Search Engine for Hidden Web that can reduce the cost of searching in term of minimizing search response time or reduced crawling time. The cost of searching can be reduced with through efficient indexing and crawling process.

Therefore, in this chapter the architecture of a Least Cost Vertical Search Engine based on Domain Specific Hidden Web Crawler (DSHWC) for Hidden Web has been proposed that solves the above mentioned issues. The proposed work consists of the following function components:

- DSHWC
- Load Balancer
- Efficient Indexing Technique
- Ranking
- Query Processing
- Next Query Prediction

The next section provides the overview of proposed Architecture.

3.2 PROPOSED ARCHITECTURE OF A LEAST COST (LC) VERTICAL SEARCH ENGINE BASED ON DSHWC

This section provides a brief introduction about the proposed architecture and an overview of the components of the Least Cost Vertical Search Engine (LCVSE). The advantage of vertical search engine is that it allows the users to collect the content from a particular domain and access the Web pages that belong to the specific topics for the Hidden Web. The vertical search engine gives results that are most relevant to the user. The components of proposed Least Cost Vertical Search Engine based on DSHWC are briefly described in Table 3.1 with their functionality.

Table 3.1: Proposed Architecture Component Description

S.No.	Component	Function / Description
1	Hidden Web Crawler DSHWC[15]	This crawler collects the contents from Hidden Web crawler.
2	Page Repository	Stores Hidden Web pages that are collected by DSHWC.
3	Indexer	Indexing the pages stored in page repository.
4	Query Interface	Query interface helps to specify the query posed by the user in natural language.
5	Query Processor	Receive a query from the query interface and process it and generate sub queries.
6	Domain Identifier	Identify the domain of the query and classify the query.
7	Term Matcher (DSIM)	Matches the query terms with the index.
8	Result Merging	Combines the matched results from different indexes and provides combined result set to reduce the redundancy.
9	Rank Calculator	Computes the rank and arranges the pages according to their importance.
10	Query Log	Contains the queries fired by different users at different time.
11	Web Log	Contains the record of actions by the users at different interval of time.
12	Frequency Calculator	Contains frequency of each query fired by different users.
13	Next Query Predictor	Predicts the next query that could be fired by a particular user.

The architecture of the proposed Least Cost Vertical Search Engine (LCVSE) based on Domain Specific Hidden Web Crawler (DSHWC) has been shown in the Figure 3.1. The working of the proposed LCVSE depends upon the various components.

The proposed architecture employs Hidden Web Crawlers that downloads the pages from the WWW effectively and efficiently and stores them in the page repository. The purpose of using multiple instances of Hidden Web Crawlers is to avoid overload of crawling task of a single crawler, and this is done with the help of load balancer [76].

The various Hidden Web Crawlers fetch the information searched by the user. The load balancer then distributes the hardware and software resources to the Hidden Web Crawler thereby reducing the search time. The proposed architecture assigns a dedicated page repository, indexer and index to each instance of Hidden Web Crawler. This reduces the communication, retrieval, searching and accessing cost. The Hidden Web Crawler uses the WWW to download the Web pages. Page repository stores and manages the large number of dynamic pages retrieved by the Hidden Web Crawler.

The dynamic pages and databases are stored in structured manner. The dynamic pages in the page repository are indexed using the indexer with the help of the Hidden Web Crawler. The Indexer creates the index for the pages that are stored in the page repository. Indexing is used to quickly access the dynamic pages and database from data source. Once the user issues a query through query interface, the end users are able to convey their information.

After receiving of a query from the query interface, the query processors processes the query and generates sub queries for further processing, it is passed through the domain identifier for quick identification of the domain. It contain the all domain covered by the search engine. This module searches the query term in the domain list and identify the domain. For example: Flight/ Delhi/ Lucknow.

After the identification of the domain Flight term is removed from the list and the final words for matching are Delhi / Lucknow. After the identification of the domain, Term Matcher plays very important role for the automatic identification of the semantic relationships between attributes of different search interfaces and also is responsible for searching the query term in the domain specific Index that is basically in the form of Schema-Instance and provides the matching attributes in form of result pages.

Thereafter, it combines the data with different sources and provides the combined view of the data to the user. The success of search engine depends upon rank of Web pages.

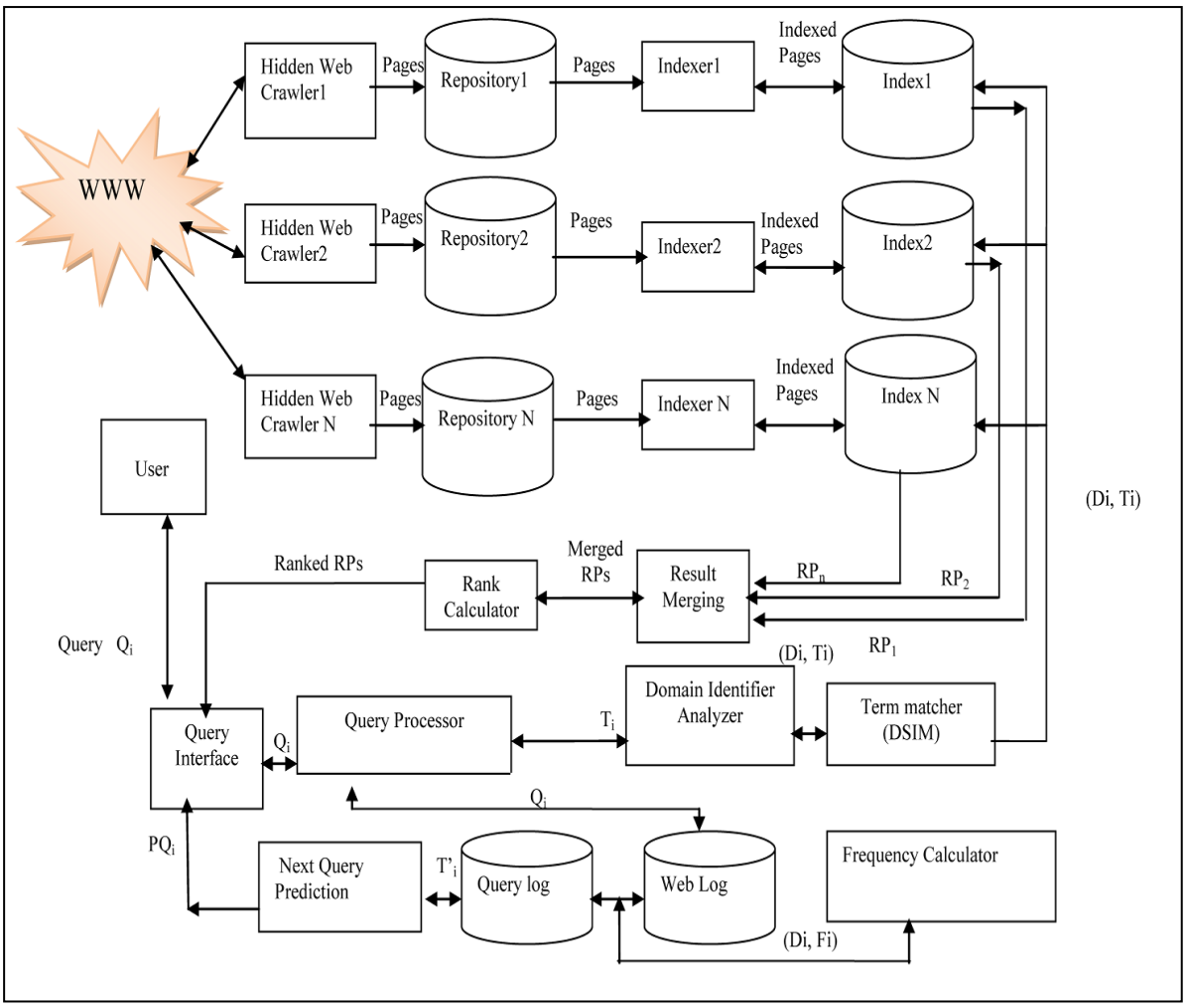


Figure 3.1: Architecture of a "Least Cost Vertical Search Engine based on DSHWC"

The Web log maintains the details of actions performed by the users. Whenever the user accesses the domain specific search engine, the Web log updates the information and frequently stores it. Further every Web server maintains the list of actions performed / requested by the user into Web log files. The collection of the queries is fired by the user stored in the query log. Query log contains the record of queries fired by different users at different time and it stores the database on the basis of the keywords of various domains and helps in the searching or firing the query.

The history of user access behaviour is stored in query log as well as in Web log. However, the Query Log only contains the fired query information or searching query. The occurrence of each query in query log is calculated by the frequency calculator. The analysis of Web log, query log and frequency of query is used by frequency calculator for the next query prediction. The next query prediction is based on the analysis of the user behaviour, preferences and frequent item set. The output of

the generated query after passing through the various components is provides the relevant information to the end user. The next section discusses the load balancer concept. The load balancer techniques directly apply on network, hardware and software. The load balancer automatically reroute the user traffic, allocates task and manages workload. The main advantages of load balancer are maximum utilization of resources and minimizing the response time.

3.3 LOAD BALANCER

As per the proposed work to design the Least Cost Vertical Search Engine Based on DSHWC a load balancer is added to further reduce the hardware and software communication and searching time. Load balancer allocates / manages workload of Hidden Web Crawlers, indexing and page repository because of maximum utilization of resources and minimizing the response time.

After adopting the load balancer technique the process work with increased reliability. Form the user point of view if any resources like Hidden Web Crawler instance fails at any movement load balancer automatically reroute the user traffic. In the proposed architecture, DSHWC uses centralized method to balance the load of the Hidden Web Crawler because every Hidden Web Crawler have dedicated indexer, index and page repository to reduce the searching, accessing time of user and maximum utilization of Hidden Web Crawler. Load balancing techniques has been directly applied on software, network, and hardware shown in Figure 3.2.

The load balancer distributes the hardware and software resources to the Hidden Web Crawler thereby reducing the crawling time. The fundamental concept of load balancer includes the node or server for the reception of network traffic mainly IP addresses of the physical server. The important features of load balancer are:

- Resources scaled up on demand
- Distributes workload
- Maximum utilization of resources
- Minimizing the response time
- Maximizing throughput
- High availability of resources

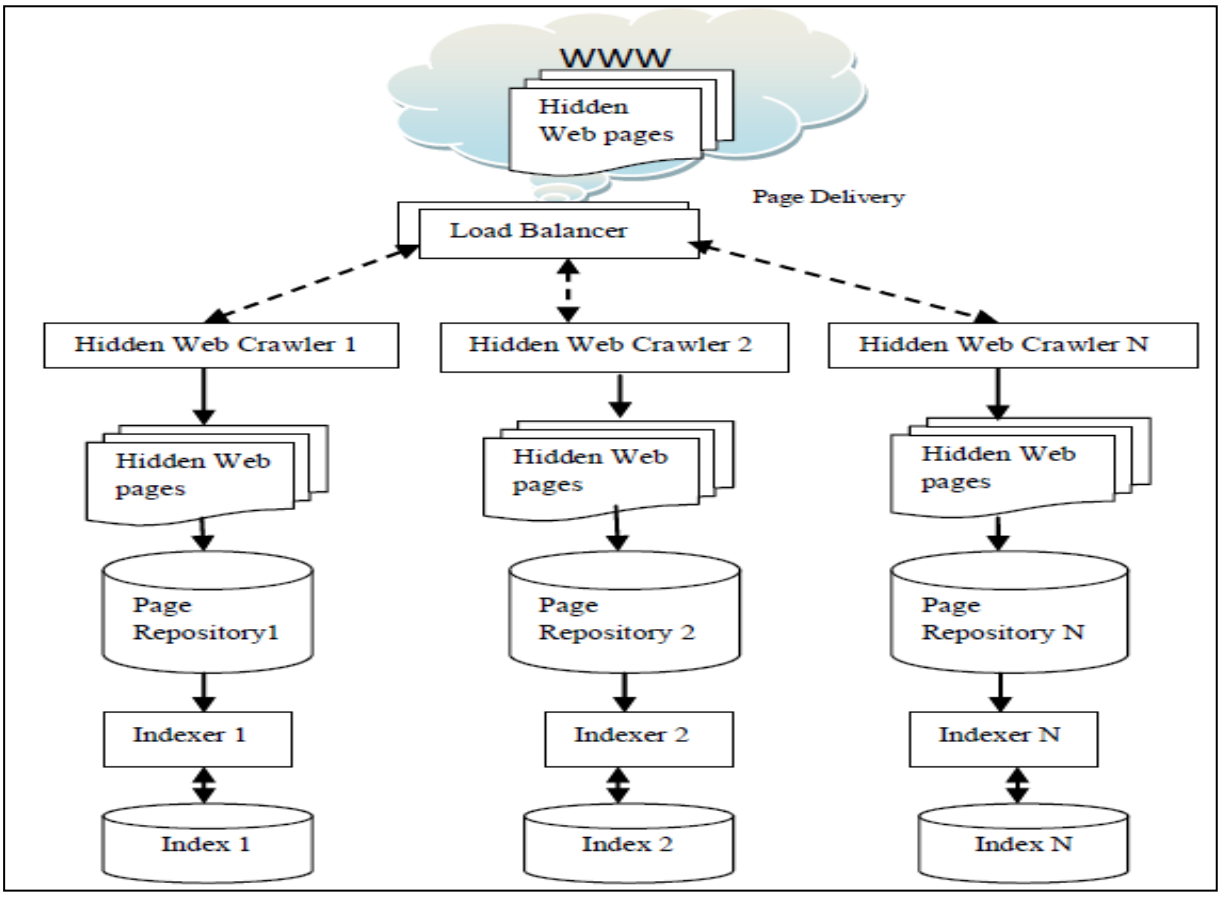


Figure 3.2 Load Balancer Mechanisms for Hidden Web Crawler

The next section discusses the components of the proposed architecture in brief.

3.4 COMPONENTS OF PROPOSED ARCHITECTURE

The brief working of each component of Least Cost Vertical Search Engine based on DSHWC has been discussed as follows:

3.4.1 Hidden Web Crawler

In this work Domain Specific Hidden Web Crawler has been used that downloads Hidden Web pages effectively and efficiently from the Hidden Web resources according to their domain and stores it in page repository. In this work, book, airline has been taken into the consideration. Detailed working of DSHWC has been discussed in chapter 4.

3.4.2 Indexer

Indexer is an important component that generates an effective index for Least Cost Vertical Search Engine with the help of Hidden Web pages that are already stored in the page repository. To do this, it generates Schema-Instance based indexing that helps the search engine to retrieve the required results correctly. The detailed working of Indexer has been discussed in chapter 5.

3.4.3 Query Interface

Query interface provides an interface to user to enter the query related to Hidden Web. The working of query interface has been discussed in chapter 6.

3.4.4 Query Processor

It receives the query from query interface in natural languages and converts the query into query term after the correcting the query forms. For example: Flight from Delhi to Mumbai. To generate query terms from the query, it applies various techniques like stop-words removal, stemming, tokenization, etc. The Output of the example: Flight /Delhi/ Mumbai.The detailed working of query processor has been discussed in chapter 6.

3.4.5 Domain Term

The query is generated for the search engine by the query processor. It contain the all domain covered by the search engine. This module searches the query term in the domain list and identify the domain. For example: Flight, Delhi, and Mumbai. After the identification of the domain Flight term is removed from the list and the final words for matching are Delhi Mumbai. The detailed working of domain term has been discussed in chapter 6.

3.4.6 Term Matcher

Term matcher is responsible for searching the query term in the domain specific index that is basically in the form of Schema-Instance and provides the matching pages in

form of result pages. The detailed working of term matching has been discussed in chapter 6.

3.4.7 Result Merging

The role of result merging combines the result from different sources and provides combined result set. The detailed working of result merging has been discussed in chapter 7.

3.4.8 Rank Calculator

The success of Least Cost Vertical Search Engine based on DSHWC depends upon rank of Web pages. Rank calculator arranges the result pages according to their relevance. The detailed working of rank calculator has been discussed in chapter 7.

3.4.9 Frequency Calculator

Frequency calculator determines how many times a particular query is repeatedly issued by different users in the given span of time. The detailed working of frequency calculator has been discussed in chapter 8.

3.4.10 Next Query Prediction

It makes the prediction about the next query that the user is going to fire. The success of next query prediction depends upon the analysis of the query fired by the particular user.The detailed working of next query prediction has been discussed in chapter 8.

The next chapter discusses the proposed Load Sharing Architecture of Domain Specific Hidden Web Crawler (DSHWC) that enhances the performance of crawling.

Chapter IV

LOAD SHARING ARCHITECTURE OF DOMAIN SPECIFIC HIDDEN WEB CRAWLER (DSHWC)

4.1 INTRODUCTION

Generally, users fire the query through the query interface and interact with the general search engine in Figure 4.1. The general purpose search engine does not work effectively for finding the topic of relevant search over the Hidden Web. The "Hidden Web" or "Invisible Web" contents are currently not a part of conventional search engine. The searching of the Hidden Web is very difficult due to the two basic reasons. The first issue is size of the content stored in online database and secondly, it requires access to the database through restricted search interface so as to extract relevant content.

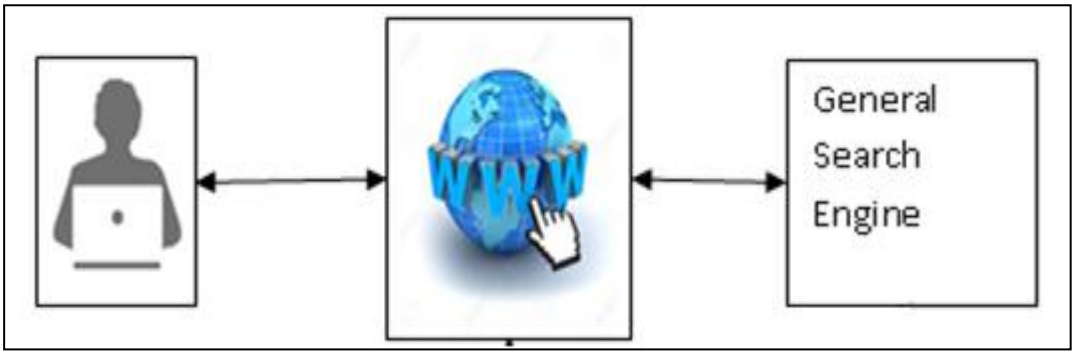


Figure 4.1: User Interaction

After the reviewing many research articles, the merits and demerits of Domain Specific Hidden Web Crawler are mentioned in the Table 4.1

Table 4.1 Hidden Web Crawler Merits- Demerits

S.No	Paper Title	Merits	Demerits
1	A Framework of Deep Web Crawler[14]	I. Efficient crawling from deep Web II. Crawler manager controls the entire crawling process III. For ranking used weight as	I. Predetermined set of target sites

		probabilities	
2	AKSHR: A Novel Framework for a Domain-specific Hidden Web Crawler[15]	I. AKSHR produces better results as compared to the results obtained from other Hidden Web Crawler	I. Specialized search engine for Hidden Web
3	Finding the WDB's Query Interface in Deep Web Automatically [16]	I. Automatic Classification of query interface	I. Accept lower case sentence only
4	A Crawler for Local Search [17]	I. Highly relevant Web pages II. Crawling and Indexing a large amount Webpage	I. More hardware required II. Fails to show up in other pages
5	A Vertical Search Engine – Based On Domain Classifier [18]	I. Page Relevancy to a particular domain, II. Keyword based search engine to improve quality of URL	I. Phrase Search II. Link analysis can be added for to provide better result
6	Crawling the Hidden Web [19]	I. Task specific Hidden Web Crawler II. Human assisted approach to crawling Hidden Web	I. Few forms are filled incorrectly
7	Exploiting Ontology for Retrieving Data Behind Searchable Web Forms [20]	I. Performance of Hidden Web crawlers in terms of precision, recall, time and cost. II. Retrieve all the data behind searchable forms	I. Link or Button based approach used

8	Hidden Web Database Exploration [21]	I. Hidden database analysis and statistics II. Parsing III. Form Identification IV. Classification of Web Hidden database	I. Predefined classification criteria II. Indexing work can be extended

As per the suggestions made by the various authors in the research articles the operation of the Hidden Web Crawlers has been described in four to five phases. The architecture of Domain Specific Hidden Web Crawler suggested by [Komal Bhatia .et. al], comprises of four phases.

1. Search Interface Downloader
2. Domain Specific interface mapping
3. Automatic form filling
4. Response page analysis

The complete framework of Domain Specific Hidden Web Crawler consists of following four unique features that differentiate it from other existing Hidden Web Crawlers:

- Automatic downloading of search interfaces to crawl Hidden Web databases
- Automatic identification of semantic relationship between attributes of different search interfaces by using Domain Specific Interface Mapper (DSIM),
- Provides the facility of automatic as well as manual filling of unified search interface
- Load balancing

The next section discusses the Load sharing architecture of Domain Specific Hidden Web Crawler (DSHWC) for Hidden Web. Load balancer technique and component of DSHWC which is used to reduce the cost of crawling, accessing along with communication cost in term of minimizing response time and maximum utilization of network bandwidth for the Hidden Web contents.

4.2 PROPOSED LOAD SHARING ARCHITECTURE OF DOMAIN SPECIFIC HIDDEN WEB CRAWLER

The proposed architecture employs number of instances for Hidden Web Crawlers. The task of Hidden Web Crawler is to deal with the automatic finding of the relevant Web site and also obtaining the information from those sites. The purpose of using multiple instances of Hidden Web Crawlers is to avoid pipelining of searching Hidden Web pages, and this is done with the help of load balancer. It downloads the pages from the WWW effectively and efficiently according to their domain and stores them in the page repository.

Web pages and extracted information searched by Hidden Web Crawler stored into the dedicated page repositories. The proposed architecture assigns a dedicated page repository; indexer and index to each Hidden Web Crawler. As per the literature review [14,15,16,17,18,19,20,21] presented above usually there are 3-5 methods of Domain Specific Hidden Web Crawler searches, however the desired information is processed in the following four phases.

In the proposed architecture the load balancer has been further incorporated in order to distribute the crawling/searching process thereby reducing the crawling time along with communication time for searching of the Hidden Web contents of a particular query. The rationale of the proposed architecture is to facilitate the user by increasing the efficiency of the whole Web search mechanism.

This is achieved through employing the various techniques that involves the automation of a large number of parameters like searching, viewing, filling in and submitting the search forms. In addition to the above response page analysis is also carried out. Due to the involvement of many complex activities, the proposed crawler employs into five phases.

1. Load Balancer
2. Search Interface Downloader
3. Domain Specific interface mapping
4. Automatic form filling
5. Response page analysis

In fact, the objective is to automate the process of searching, viewing, filling in and submitting the search forms followed by analysis of the response pages. Since the proposed crawler involves many complex activities, five phases has been designed to separately handle the various groups of actions. Each phase is designed to perform specific group of actions distinctively. A brief discussion of each phase is given below:

4.2.1 Load Balancer

Load balancer allocates / manages workload to Hidden Web Crawlers. The purpose of load balancer is maximum utilization of resources and minimizing the response time. For the user point of view if any resources like Hidden Web Crawler instance fails at any movement load balancer automatically reroute the user traffic. Load balancing techniques directly apply on network, hardware and software. The working of the load balancer shown in Figure 4.2

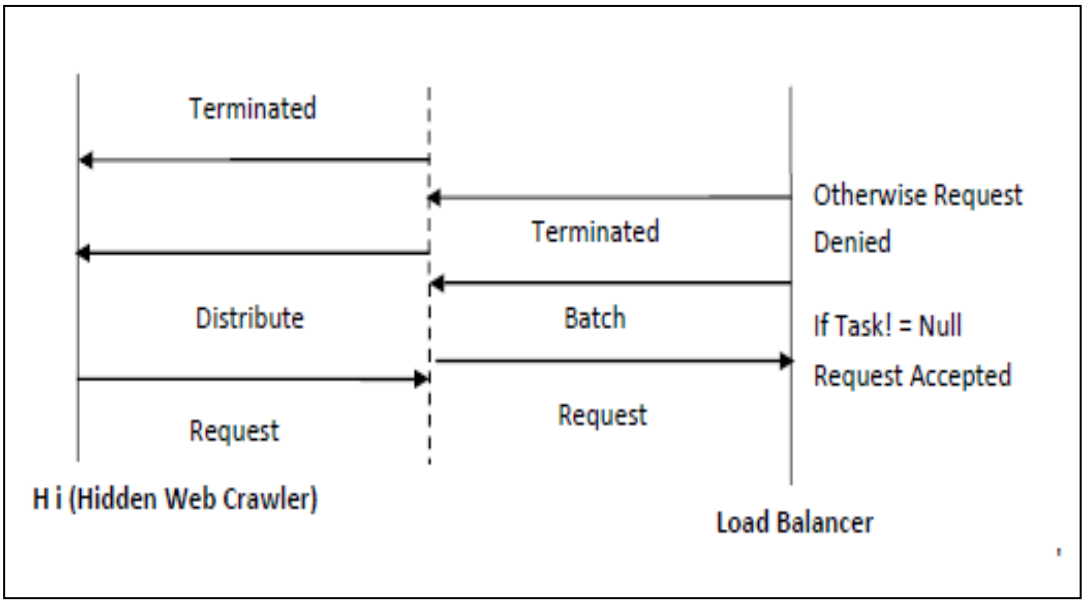


Figure 4.2: Working of Load Balancer

The proposed Load Sharing Architecture of Domain Specific Hidden Web Crawler is shown in Figure 4.3.

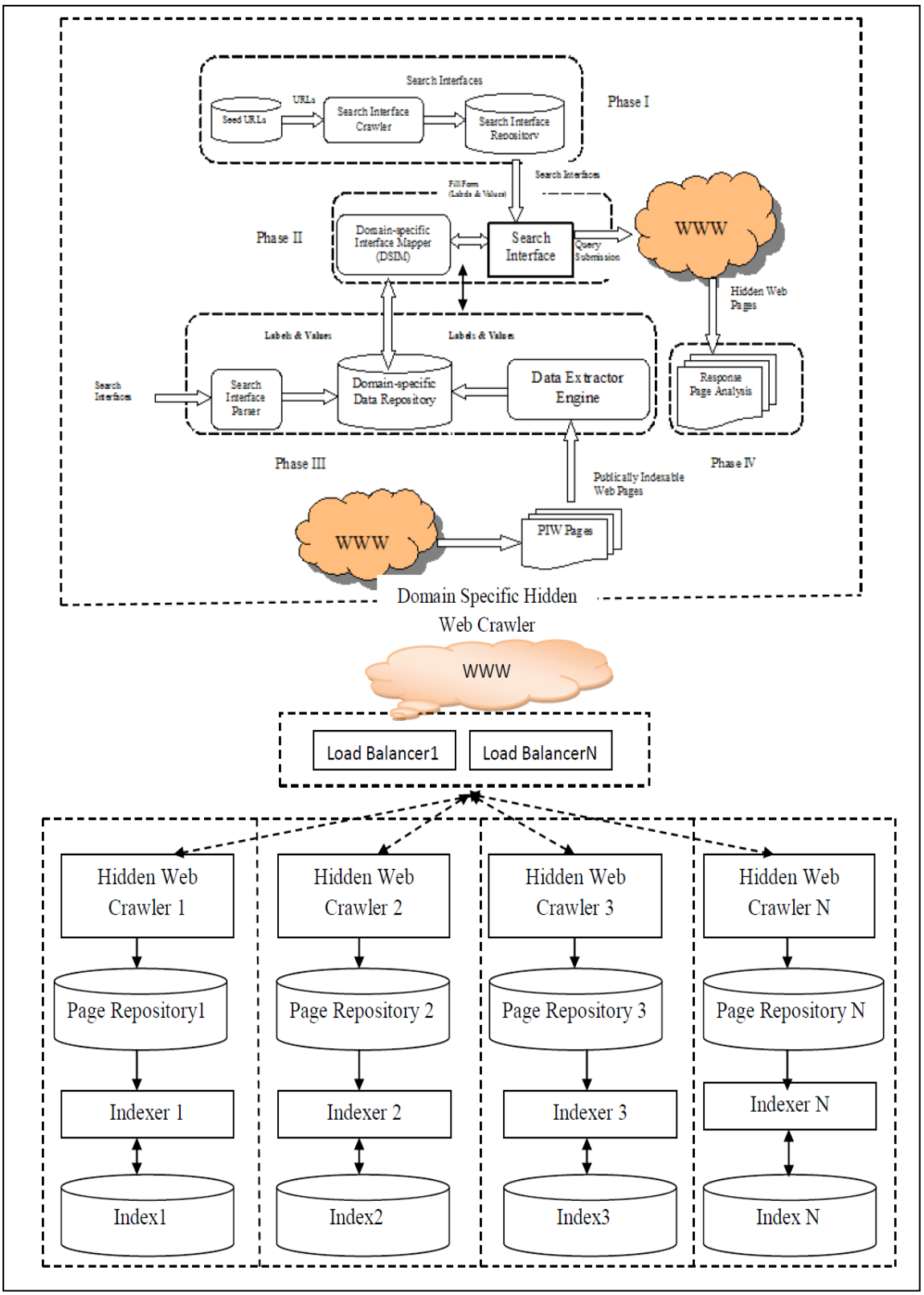


Figure 4.3: Load Sharing Architecture of DSHWC

The load balancer distributes the hardware and software resources to the Hidden Web Crawler thereby reducing the crawling time. The proposed algorithm Figure 4.4 for load balancer given below:

```
Algorithm Load Balancer ()
Begin
    Step1: Request message received from the Hi (Hidden Web Crawler)
    Step 2: The load balancer accepts the connection
            If (task! = Null)
    Step 3: Perform task and then ask for another task
    Step 4: Send request message to the load balancer
    Step 5: Otherwise termination message received from   load balancer
End
```

Figure 4.4: Algorithm of load balancer

4.2.2 Search Interface Crawling

The second phase of the crawler is known as the search interface crawling. The search interface acts as a link which facilitates the user to access the crawler and provides the interface to the Hidden Web. Another way to enhance the efficiency is to apply the parsing technique only to those Web pages acting as search interface. This is further classified in to three main steps:

1. Seed URL
2. Search interface Repository:
3. Search Interface Crawler

This phase employs the tasks like collection of Web pages, after which the indexing of the same follows. This enables the entry of the proposed Web crawler. Such type of crawler is known as the search interface crawler.

4.2.3 Domain Interface Mapping

In the third phase, the architecture employs the task that automatically identifies or detects the semantic mappings between the attributes of the search interfaces.

However, the interface mapping finds the necessary semantic correspondences of attributes across Web interfaces. Also the merit of interface mapping lies when the domain specific applications have to search for the alternative resources of data in the same domain. So, it can be interpreted that it involves the merging of collected search interfaces with the unified search interface (USI). In order achieve this merging; semantic mappings are stored in Domain Specific Mapping Knowledge base.

4.2.4 Automatic Form Filling

In the fourth phase, the automation of the task to fill the Unified Search Interface is carried out, though USI can be filled manually. A Domain Specific Page Repository consisting labels and their corresponding values has been employed to automatically fill the Unified Search Interface (USI) which is later on submitted. The attributes of the Unified Search Interface (USI) are matched with the attributes of Domain Specific Page Repository and correspondingly the values are extracted to fill the search interface. Thereafter the USI is submitted to specific sites to obtain the desired results.

4.2.5 Response Pages

The last phase analyzes the response pages with a view to shift the erroneous pages. The response analyzer differentiates among the response pages consisting of the search results, and pages containing error messages. The pages containing error messages illustrate that no matches were found against the fired queries while the pages containing search results illustrate that information was found against the fired queries. The purpose of the Hidden Web Crawler is to crawl large amount of high quality information 'Hidden' behind search forms. It automates the downloading of search interfaces, finds the semantic mappings, merges them into Unified Search Interface (USI) and fills the USI thereof. Finally, the Crawler submits the form to obtain the response pages from the Hidden Web. The load balancer manages the results from the proposed architecture which comprises of Hidden Web Crawlers instance by linking of the respective pages containing the desired results thereby reducing the time of searching and computation.

The next section discusses the proposed expected response time computation method.

4.3 PROPOSED EXPECTED RESPONSE TIME COMPUTATION METHOD

The cost of crawling depends upon the size of database and the communication rounds between the crawler and the Web server. Cost measured in terms of response time. The response time is measured as follows:

- Time spend in submitting the query at server
- Time spend in retrieving the result
- Receive time from server to client

If there are 20 records used in database matching the attribute value "Author Name" and each result page displays the next 15 Records, the total cost to retrieve the entire answer set will be 15/20. This can be obtained in two communication rounds. Results are based on the type of query generated. For example if query is fired about the specific result like search on subject history shown in Figure 4.5 and three records are obtained then the final value records crawled is 3/20 records.

Result page

Author's First Name	Author's Last Name	Title	ISBN	Publisher	Subject	Keywords	Buy now
Levi	Tillemann	The Great Race: The Global Quest For The Car Of The Future	1476773491	Simon & Schuster	History	agined. They will drive themselves, won't consume oil, and will come in radical shapes and sizes. But the path to that future is fraught The top contenders	Buy now
Mike	Mueller	The Complete Book Of Corvette - Revised & Updated: Every Model Since 1953 (Complete Book Series)	0760345740	Motorbooks	History	flagship sports car has become a timeless part of American culture and a household name across the	Buy now
A. J.	Baime	Go Like Hell: Ford, Ferrari, And Their Battle For Speed And Glory At Le Mans	0547336055	Mariner Books	History	omobile transportation to the masses was falling behind. Baby boomers were taking to the roads in droves, looking for speed not safety style not comfort	Buy now

Figure 4.5: Search Result (Subject History)

This is because each result page can typically hold a fixed number of matched records and thus every initiated connection retrieves at most data records. The crawling cost (Q*i*, PR*)* of querying the page repository (PR) with query (Q*i*) is defined as: cost (N,*PR*)= where N stands for the number of matched records in PR and num (N,PR) / *R* corresponds to the maximum number of records displayed in each result page. Through the parallel computation speedup increased significantly. As per the Table 4.2 the time taken to retrieve the information by single crawler is 0.0019s for 20 records where all values are retrieving with general query, for five records it is 0.0020s

for specific results and for 20 records it is 0.0022s according to the information retrieving by bookid.

Table 4.2: Actual Completions Time

No. of Process	Matched Record	Response Time	Searching Method	Domain	Total Records
1	20	.0019	All Information	Book	20
1	5	.0020	Specific Information	Book	20
1	20	.0022	Condition based Information	Book	20

To increase the efficiency and reliability of the system, the architecture of the proposed Least Cost Vertical Search Engine based on DSHWC proposed multiple instances of Hidden Web Crawler with dedicated page repository and indexing mechanism employed for obtaining relevant and fast results. It is managed and controlled by the load balancer. The computation proposed formula given below.

$$Total\ number\ of\ matched\ record = N ---------- eq. 4.1$$

$$The\ time\ to\ complete\ by\ single\ crawler = NT ------ \quad eq.\ 4.2$$

$$crawler\ be\ employed = K ---------- \quad --eq. 4.3$$

$$Wait\ time\ of\ crawler = Q, Where\ Q\ is\ negligible --- \quad eq. 4.4$$

$$Actual\ response\ time\ by\ single\ crawler = [NT + Q]-- \quad eq.\ 4.5$$

$$The\ expected\ time\ to\ complete\ by\ K = \frac{[NT+Q]}{K} ------ eq. \quad 4.6$$

The proposed formula based calculation (Expected Response Time) is shown in Table 4.3.

Table 4.3: Expected Response Time

No. of Process	**Matched Record**	**Expected Response Time**	**Searching Method**	**Domain**	**Reuter Collection**
2	20	.000095	All Information	Book	20
2	5	.0010	Specific Information	Book	20
2	20	.0011	Condition based Information	Book	20
5	20	.000038	All Information	Book	20
5	5	.00040	Specific Information	Book	20
5	20	.00044	Condition based Information	Book	20

The next chapter discusses the Indexing of Hidden Web contents for efficient retrieval in details.

Chapter V

INDEXING OF HIDDEN WEB CONTENTS FOR EFFICIENT RETRIEVAL

5.1 INTRODUCTION

In the previous chapter, the working of the Domain Specific Hidden Web Crawler was discussed. The Hidden Web Crawler downloads the contents from the Hidden Web. This chapter proposes an indexing mechanism that is efficient for Hidden Web contents. Index structure for efficient retrieval of information for general / visible web for many indexing techniques has been proposed by researchers shown in Table 5.1; each technique has its own advantages and disadvantages.

Table 5.1: Indexing Techniques

S.No	Paper Title	Indexing Techniques
1	Performance Optimizations for Distributed Real-time Text Indexing [26] (Ankur Narang et.al)	i. Interval Hash Table ii. Global Hash Table based Index
2	Highly Scalable Algorithm For Distributed Real-Time Text Indexing [27] (Ankur Narang et.al)	i. Interval Hash Table ii. Global Hash Table based Index iii. index merge process
3	How Search Engines work and a Web Crawler Application [13] (Monica Peshave et.al)	i. Key word Indexing ii. Full Text
4	Research and Improvement of Search Engine based on Lucence [28] (Young Zhang et.al)	i. Pretreatment ii. Analyzing Text iii. Index Sorting
5	Research and Design of an efficient Chinese Indexing System [29] (LiWen et.al)	i. Word segmentation Algorithm ii. Index Writer Algorithm iii. Index Processor Algorithm

The proposed architecture of Least Cost Vertical Search Engine based on DSHWC (Figure 3.2) assigns a dedicated page repository, indexer and index to each Hidden Web Crawler. The Hidden Web Crawler uses the page repository to download the Web pages. The Table 5.2 shows the database of Books domain.

Table 5.2: Books Domain Database

S.No.	Author Name	Title	ISBN	Publisher	Subject	Keyword
1	Charles	Introduction to Algorithm	8120340078	PHI	Computer Science	Algorithm sorting
2	Andy Tanenbaum	Computer Network	9332518742	Pearson	Computer Science	Physical data link access control
3	Andy Tanenbaum	Modern Operating system	8120339045	PHI	Computer Science	Process threads memory
4	Willam	Operating System	9332518807	PHI	Computer Science	Memory management
5	Siberschatz	Operating System Concepts	8126520515	Wiley	Computer Science	Deadlock system
6	Andrew	Modern compiler Implementatio n in C/ Java	8131720470	Cambridge University Press	Computer Science	Scheduling Optimization
7	Hopcroft	Introduction to Automata Theory	8131720470	Pearson	Computer Science	Computational complication
-	-	-	-	-	-	-
-	-	-	-	-	-	-
20	Mike Muller	The complete Book of Corvette	076034570	Motor book	History	Flagships sports cars has become

Page repository stores and manages the large number of dynamic pages retrieved by the Hidden Web Crawler and database store in structured manner in form of a table having columns and rows. The dynamic pages in the page repository are indexed using the indexer with the help of the Hidden Web Crawler.

Therefore in this chapter, a novel Schema-Instance based indexing mechanism is being proposed that indexes the Hidden Web data. This technique uses the Schema-Instance based data representation scheme specifically for Hidden Web. The next section discusses the proposed Schema-Instance based indexing mechanism in detail.

5.2 PROPOSED SCHEMA-INSTANCE BASED INDEXING MECHANISM

The Schema- Instance based indexing refers to the piece of information in the form of column and rows (Table), attributes taken as the origin of Hidden Web documents for a particular domain[72]. The various domains like Sports, Flight, Education, Health, Books, Mass Media shown in Table 5.3 each domain contains the Domain ID and attributes of that particular domain.

Table 5.3: Domain and Domain ID Identifiers

S.No	Domain	Domain ID
1	Sports	D1
2	Flight	D2
3	Education	D3
4	Health	D4
5	Books	D5
N	Mass Media	Dn

The Books domain attributes contains Author Name, Title, ISBN, Publisher, Subject and Keyword. For example Books domain contain various attributes and corresponding attribute ID is assigned to each attribute shown in Table 5.4.

Table 5.4: Attribute and Attribute ID Identifiers (for D5)

S.No	Attribute	Attribute ID
1	Author name	A1
2	Title	A2
3	ISBN	A3
4	Publisher	A4
5	Subject	A5
6	Keyword	A6

The values of Title attribute (consisting of Books Title) like Introduction to Algorithm, Computer Network, Operating System, Modern Compiler Implementation, and Introduction to Automata Theory. The value contains value ID like V1, V2, V3, V4 and V5 respectively shown in Table 5.5.

Table 5.5: Value and Value ID Identifier (for attrID=A2)

S.No	Value	Value ID
1	Introduction to Algorithm	V1
2	Computer Network	V2
3	Operating System	V3
4	Modern Compiler Implementation in C/Java	V4
5	Introduction to Automata Theory	V5

The value ID of the Title attribute is stored in a cluster like CL1, CL2 and CL3.The CL1, CL2 and CL3 clusters containing values (V1, V5), (V2, V4) and (V3) respectively shown in the Table 5.6

Table 5.6: Value ID & Cluster ID (for attrID=A2)

S.No	Value ID	Cluster ID
1	V1	CL1
2	V2	CL2
3	V3	CL3
4	V4	CL2
5	V5	CL1

The documents contain different values in different clusters i.e. now CL1, CL2, CL3 contains (DS1, DS3, DS4, DS8), (DS1, DS2, DS5) and (DS4, DS6, DS8) respectively. Clustered index structure shown in Table 5.7 for attrID=A2.

Table 5.7: Clustered Index structure (for attrID=A2)

S.No	Cluster ID	Data Sources
1	CL1	DS1,DS3,DS4,DS8
2	CL2	DS1,DS2,DS5
3	CL3	DS4,DS6,DS8

The overall index structure shown in Figure 5.1 for attrID=A2.

S.No	Domain	Domain ID
1	Sports	D1
2	Flight	D2
3	Education	D3
4	Health	D4
5	Books	D5
N	Mass Media	Dn

Attribute	Attribute ID
Author Name	A1
Title	A2
ISBN	A3
Publisher	A4
Subject	A5
Keyword	A6

Value	Value ID
Introduction to Algorithm	V1
Computer Network	V2
Operating System	V3
Modern Compiler Implementation in C/Java	V4
Introduction to Automata Theory	V5

Data Sources	Cluster ID
DS1,DS3,DS4,DS8	CL1
DS1,DS2,DS5	CL2
DS4,DS5,DS6,DS8	CL3

Cluster ID	Value ID
CL1	V1
CL2	V2
CL3	V3
CL2	V4
CL1	V5

Overall Index Structure for Book Domain: for D5 and for A2

Figure 5.1: Overall Index structure

The domain identifier (domID) is represented as D1,D2,D3,D4,D5,DN and attributes of D5 are assigned the attribute identifier (attrID) as A1, A2… A6. For each attribute there exist a huge amount of values, like A2 have V1, V2, V3, V4, V5 values, and the values of attributes are divided into clusters. In this case, V1, V5 belongs to in clusters CL1; V2, V4 belongs to in cluster CL2 and V3 belongs to in cluster CL3. The documents contain different values in different clusters i.e. now CL1, CL2, CL3 contain (DS1, DS3, DS4, DS8), (DS1, DS2, DS5) and (DS4, DS6, DS8) respectively.

The next section discusses the proposed algorithm for indexing like index construction and index search in detail.

5.3 PROPOSED ALGORITHM FOR INDEXING

The proposed index construction algorithm classifies the documents according to the various domains and assigns the domain IDs in accordance with the attribute value. In next step is to identify the attributes, identify the values and allocate the values and thereafter the values are stored in the cluster in different data sources [72]. To do this it generates attribute value pair that helps the search engine to retrieve the required results correctly. The algorithm for index construction is given in Figure 5.2

Algorithm Index Construction ( )

{

Step1: Classify the documents according to the domains

Step2: Assign domID to each domain

Step3: Identify the attributes of different domains from Unified search interface of that domain and assign attrID to each attribute in the index structure.

Step4: Create the clusters of documents depending upon the value of each attribute.

}

Figure 5.2: Algorithm for Index Construction

The Indexer reads the user query and then identifies the domain and its domain ID. Thereafter, it identifies the attributes of the particular domain and its attribute ID. The data is then retrieved on the basis of attribute value pair. Finally the cluster ID of each value is retrieved from the repository.

According to the proposed search structure of the indexer the speed of searching would improve by the Index Search algorithm. The algorithm for searching the index is given in Figure 5.3.

```
Algorithm Index Search ( )
{
Read a user's query                    // Identify the domain and its domain ID
dom=domain
domID=domain ID                         // Identify the attributes and their values
attr[ ]=attributes
for each attribute in attr[ ]           // retrieve the attribute ID
attrID[ ]=attribute ID                 // retrieve the value
val[ ]=value
for each attributeID in attrID[ ]      // LOOK UP INDEX
for value in val[ ]                    // retrieve the clusterID
cID[]=cluster ID
for each cID in cID[ ]                 / retrieve the data sources
list_of_ds=data sources
return list_of_ds
}
```

Figure 5.3: Algorithms for Index Search

The next section discusses the proposed working principle of proposed searching algorithm in detail with example.

5.4 WORKING PRINCIPLE OF SCHEMA-INSTANCE BASED INDEXING

The Unified Search Interface consists of the books domain with attributes like Author Name, Title, ISBN, Publisher, Subject and Keyword. After the filling of attribute value according to the fields, the user click the submit button and get the output regarding his/her queries. Suppose a user fills only the title of the book, all the records related to the title would available for the user.

To search a Books with "Title', the index maintained would be searched to give a list of Hidden Web documents i.e. DS1, DS2, DS3. Figure 5.1shows that the following contents are the attributes of the book domain. These attributes are assigned unique attribute IDs in the form of the alphanumeric values. This helps to extract the labels of

each global query interface quickly and thereby saves the search time and cost of the user. For example user fired a query: Retrieve book with Title as ‘Computer Network’

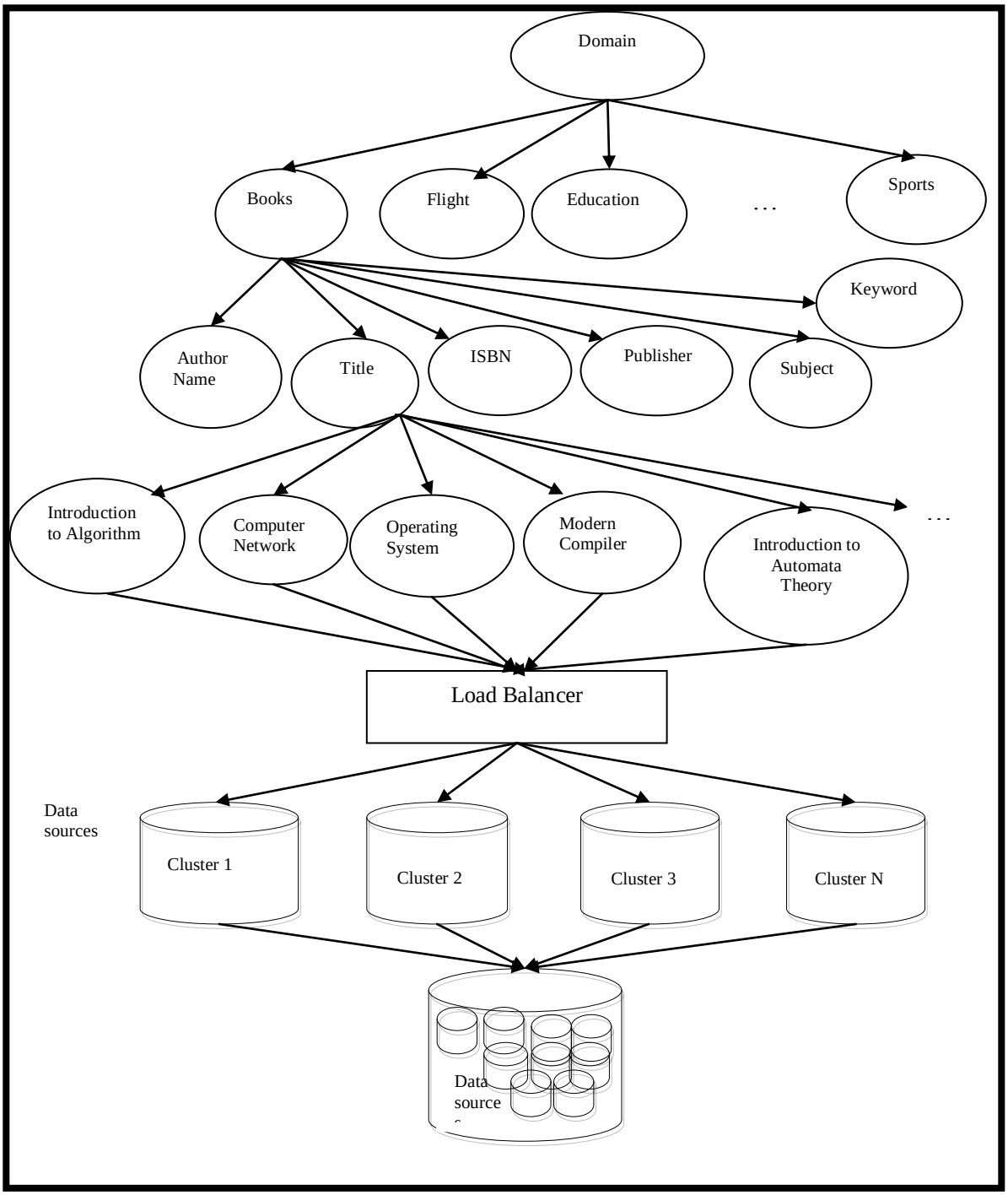


Figure 5.4: Proposed Search Structure of Hidden Web Documents

The following steps are required to retrieve the book with Title as Computer Network. This can be done by the proposed search algorithms and proposed search structure of Hidden Web document shown in Figure 5.4. Initially the stop-words, stemming and tokenization methods are apply on the fired query. Through these methods the

common words like the “Retrieve”, “with” and “as” are considered as stop words are removed from the query. After the removal the stop words the query contains the terms like “Books”, “Title” and “Computer Network”. The first step is to identify the domain out of these terms. The keyword “Books” identifies that the query is related to “Books” domain.

Figure 5.4 shows the different domains of search that can be selected by any user at any point of time. It can be observed from the table that a specific domain ID is allocated against every single domain for example domain ID of Books domain is D5.The corresponding attribute ID assigned to Title attribute is A2.Now a match corresponding to A2 is found for Computer Network in the index.

A set of documents corresponding to the cluster of “Computer Network” will be returned as the result. The list of data sources obtained above indicates the Hidden Web documents that have already been crawled and stored in Hidden Web repository. Each attribute has now been allocated there original values for example the Title of the book is given the value which is the Computer Network that is the name of the book.

The value set of all the attributes discussed above is assigned a specific Value ID. For instance the Value ID for each value of value set of Global Query Interface is listed in Figure 5.1. Assign a value from value set to each attribute of the global query interface and submitted the form.

The downloaded document corresponding to the form is stored in repository. The function of index lookup is performed by the indexer module. The domain specific working of search structure of Books domain shown in Figure 5.5.

As per the algorithm (in Fig.5.3)

dom=”Book”, (obtained by semantically mapping the identified domain with the listed domain names in the field “domain” in Fig. 5.1).

domID=”D1”, (using Fig. 5.1))

attr[]={“Title”},(obtained by semantically mapping the identified attributes with the listed attribute names in the field “attribute” in Fig. 5.1).

attrID[]={“A2”}, (using Fig. 5.1)

val[]={“Computer Network”}, (using Fig. 5.1)

CLID[]={“CL2”}, (using Fig.5.1)

For CLID=”CL2”:

list_of_ds=d2, d3, d4, d6. (using Fig. 5.1)

Figure 5.5: Domain Specific Index Search

The result in the form of table shown with attributes like Author Name, Title, ISBN, and Publisher mentioned in the Table (5.8).

Table 5.8: Books Search Result

S.No.	**Author Name**	**Title**	**ISBN**	**Publisher**	**Subject**	**Keyword**
1	Andy Tanenbaum	Computer Network	9332518742	Pearson	Computer Science	Physical data link access control

The next section discusses the comparison with keyword indexing technique in detail.

5.5 COMPARISON BETWEEN PROPOSED SCHEMA-INSTANCE AND TRADITIONAL INDEXING

The comparison between proposed Schema-Instance based indexing and traditional keyword indexing technique is shown in Table 5.9 with a query example. The

advantage of adopting Schema-Instance based indexing mechanism are increased search speed, reduced cost, high accuracy and improve qualtity of indexing.

Table 5.9: Comparison Chart

S.No	**Parameters**	**Proposed Schema-Instance based Indexing**	**Keyword based Indexing**
1	Input	Table	Document
2	Output	Schema-Instance based Index	Inverted Index
3	Technology	Structured • Indentify Domain • Identify Attributes • Indentify Value	Unstructured • Tokenization • Stemming • Stop-word
4	Redundancy in document	Low Reason: Low redundancy due to the Schema-Instance (Attributed-Value pair) based indexing technique is used.	High Reason: Keyword based Indexing used inverted index technique.
5	Searching Method	Schema-Instance (Attribute-Value pair based result)	Inverted Index
6	Quality	High	Low

The next chapter discusses the Domain Specific Query Processing in details.

Chapter VI

DOMAIN SPECIFIC QUERY PROCESSING

6.1 INTRODUCTION

The working details of the proposed indexing technique have been discussed in the previous chapter. The algorithms for constructing and searching the index were proposed. For constructing the Index following steps are performed.

i. Classify the documents according to the domains assigned domID to each domain
ii. Identify the attributes of different domains
iii. Assign attrID to each attribute in the index structure and create the clusters of documents depending upon the value of each attribute.

For searching the index following steps are performed:

i. Identify the attributes and their values
ii. Retrieve the attribute ID
iii. Retrieve the value
iv. Retrieve the corresponding cluster
v. Retrieve the data source (DS).

For example overall structure of flight domain attribute value based indexing as presented in Figure 6.1.

- Flight domain has links to attributes From, To, Class and Airline shown in the Figure 6.1
  - Attribute (From) has links with the values Delhi, Mumbai, Lucknow, Goa, etc Value ID V1, V2, V3 and V4.
    - CL1, CL2, CL3 are different clusters containing values (V1), (V2, V4) and (V3) respectively.
  - Cluster ID has links with the data sources DS1, DS2, DS3, DS4, DS5, DS6, DS7 and DS8.
    - CL1containing data source DS1, DS3, DS4 and DS8
    - CL2 containing data source DS1, DS2 and DS5
    - CL3 containing data source DS4, DS5, DS6 and DS8

The first step is to identify the domain and subsequently each domain is assigned a unique ID. Thereafter, each domain has a set of distinct attributes allocated unique attribute ID. The working principle of query processing is based on Schema-Instance based indexing mechanism. The merits of Schema-Instance based indexing mechanism are that it reduces searching time, data retrieval and accessing time. In case of flight domain, it would only have to search for just a single unique value that is assigned to the domain Flight, thereby, reducing the time of search. Similarly the "Value" table which contains the value of attributes is also assigned a unique ID. Finally these values are stored in the various clusters with further unique cluster ID. This leads to the significant reduction in the cost of searching.

Sl.No	Domain	Domain ID
1	Sports	D1
2	Flight	D2
3	Education	D3
4	Health	D4
5	Books	D5
N	Mass Media	Dn

Attribute	Attribute ID
From	A1
To	A2
Class	A3
Airline	A4

Value	Value ID
Delhi	V1
Mumbai	V2
Lucknow	V3
Goa	V4

Data Sources	Cluster ID
DS1,DS3,DS4,DS8	CL1
DS1,DS2,DS5	CL2
DS4,DS5,DS6,DS8	CL3

Cluster ID	Value ID
CL1	V1
CL2	V2
CL3	V3
CL2	V4

Overall Index Structure for Flight Domain: for D2 and for A1

Figure 6.1: Overall Index Structure of Flight Domain

Once the user issues a query through query interface the end users are able to convey their information. After receiving of a query from the query interface, indexing is first approach for improving query performance and query processor processes the query

and generates sub queries for search engine. The next section discusses the proposed work flow of query processor in detail.

6.2 WORK FLOW OF QUERY PROCESSING

When the user fires the query retrieving the information about all the flights from Delhi to Lucknow, The query is generated for the search engine by the query processor. In query processing, first step is linguistic modules, syntax checking and query modification. It then matches search term in the query syntax with views, relation and columns listed in the system relation and the system validates that the user have appropriates privileges and the query does not disobey any relevant integrity constraints. Domain Term contain the all domain covered by the search engine. The module searches the query term in the domain list and for quick identification of the domain. For example: Flight/ Delhi/ Lucknow. After the identification of the domain Flight term is removed from the list and the final words for matching are Delhi / Lucknow. After the identification of the domain, Term Matcher plays very important role for the automatic identification of the semantic relationships between attributes of different search interfaces and also is responsible for searching the query term in the domain specific Index that is basically in the form of Schema-Instance and provides the matching attribute in form of result pages shown in Figure 6.2

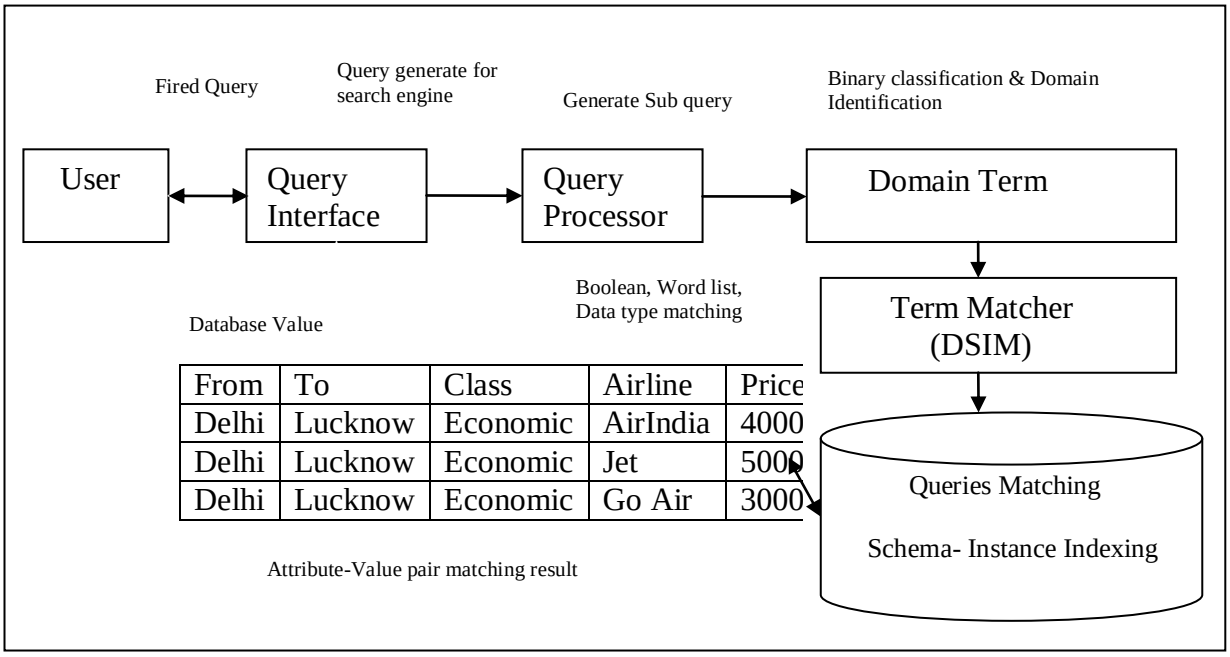


Figure 6.2 Work Flow Diagram of Query Matching

Thereafter, the extracted domain attributes are fetched by the search engine. The search engine is used to download hidden web pages effectively and efficiently from the Hidden Web resources according to their domain stores it in page repository. The next section provides the algorithm for the query processing.

6.3 QUERY PROCESSING ALGORITHM

The working of the query processor is divided into the five steps. The algorithm is explained with example for Flight domain.The algorithm of query processing shown in Figure 6.3

Algorithm Query Processing()

{

Step 1: Apply Linguistic modules such as Tokenization, Stop-word removal, Lemmatization and Normalization before processing the query

Step 2: Identify the domain from the query after applying step 1 by comparing each keyword with the list of domains.

Q = Query fired by user

$Q=\{K_1, K_2,K_3,K_4,--,--,--,--,K_n\}$

K_i is the keyword that is identified the domain

Step 3: Delete the keyword that helped to identify the domain from the Query.

$Q'= \{K_1, K_2,K_3,K_4,--,--,--,--,K_n\}- \{ K_i\}$

Step 4: Search the Q in the attribute value based index.

Step 5: Represent the result in the form of structure manner (Table).

}

Figure 6.3: Algorithm Query Processing

For example, user fired a query to search any Flight from Delhi to Lucknow through Search Interface shown in Figure 6.3.

Q = Query fired by user

Q = {Flight from Delhi to Lucknow}

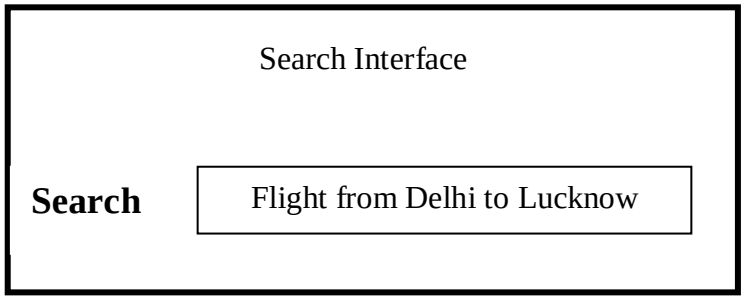


Figure 6.4: Search Interface

Step1: Apply Linguistic modules such as Tokenization, Stop-Word Removal, Lemmatization and Normalization before processing the query .The three basic phases are performed in step 1: Tokenization, Stop-Words Removal, and Stemming.

Phase 1: Tokenization:

The first step is tokenization. In this step the phrase is divided into an atomic word. In other words sentence is divided into the chunks according to the boundary condition. Though this process document is divided into the individual words and also removes the punctuations. The text represents a boundary condition with symbol like'-'. The examples of the tokenization are:

Input: Flight, from, Delhi, to, Lucknow

Output: Flight from Delhi to Lucknow

Phase 2: Stop-Words Removal

Through this step, the most common words and irrelevant are removed from the query because in the searching of the relevant web pages common words cannot decide documents. This is due to the fact that the common words appearing in every document, for example, words such as 'and', 'who', 'the', etc are very frequent. The presence and absence of the common words do not influence the result. These common words and lexical words for example 'want' 'are not helpful for searching. Therefore the words are not useful for searching it is known as Stop-Words and are completely removed from the

Index. The most frequent used Stop-Word around 500 words. (See in Table 6.1)

Table 6.1: Stop-Word List and Advantages

Stop-Word List	Advantage
a, able, about, across, after ,all, almost, am, among, an, and, any, are, as, at, be, because, been, but, by, can, cannot, could, did, do, does, either, else, ever, every , for, from, get, got, had, has, have, he, her, hers, him, his, how, however, i , if, in, into, is, it, its, just, least, let, like, likely, may, me, might, most, must, my, neither, no, nor, of, off, often, on, only, or, other, our, own, rather, said, say, says, she, should, since, so, some, than, that, the, their, them, then, there, these, they, this, to, too, us, want, wants, was, we, were, what, when, where, which, while, who, whom, why, will, with, would, yet, you, your	i. Reduce indexing file size (20-30%) ii. Improve efficiency iii. Reduce searching cost and time iv. Reduce storage space and cost

The stop-words in the given example are:

Query Input: Flight from Delhi to Lucknow: It contains two stop words (From, To)

In the removing process, the system checks the list of the stop-words. Once this process is over, all the stop-words are removed. After the removing stop-words the query terms are: Flight Delhi Mumbai

Phase 3: Stemming

Stemming is very useful to identify the correct form of the words, because words may contain a word in many different forms. For example, learn may be present as ’learn’, ’learning’, ’learned’. In all forms, the concept of learning is present. Concept is important; however, the form is not important. Such types of conditions are handled by Stemming process. All types of forms of each word are converted to their root form by removing suffix and prefix. The

Porter Stemmer and Affix Stemmer method and advantages are shown in Table 6.2.

Table 6.2: Stemming Method and Advantages

Method	Rules
Porter Stemmer Method (Techniques used to find out the root/ stem of a word)	Remove suffix • If a word ends with a constant other than s, followed by s, then delete s. • If a word ends in es, drop the s. • If word ends in ing, drop ing. • If word ends with ed, drop ed. • If word ends with ly , remove ly. Transform word • If a word ends with 'ies' consider as y.
Affix Stemmer	Remove Prefix • Indentify the leading 'in' is a prefix can be removed.
Advantage	• Improve effectiveness • Reducing Indexing size (40-50%) • Reduce searching cost and time • Reduce storage space and cost

Step 2 (Domain Term): Identify the domain from the query after applying step 1 by comparing each keyword with the list of domains.

Q={K_1, K_2,K_3,K_4,--,--,--,--,K_n} For example, { Sport, Cricket, Tennis, Football}

K_i is the keyword that is identified the domain

Q = {Flight Delhi Lucknow}

Domain Term contain the all domain covered by the search engine. The module searches the query term in the domain list and for quick identification of the domain. The second phase is to identify the domain on the basis of the keywords and matching of domain shown in Figure 6.5(a) & Figure 6.5(b). The domain is identify after matching the keywords, which are used in the query after the processing of the phase one output of the query is converted into only three words i.e Flight Delhi Lucknow.

Domain Identification List

- Sports: Cricket, Tennis, Football,----------
- Flight: From, To, Airline,------
- Education: Course, Professor, University,----------
- Health: Diet, Treatment, Doctor,----------
- Book: Author Name, Title, ISBN, Publisher,----------
- Mass Media: News, Television, News paper,----------

Figure 6.5(a) Domain Identification List

The hierarchical representation of domain URL shown in Figure 6.5(b)

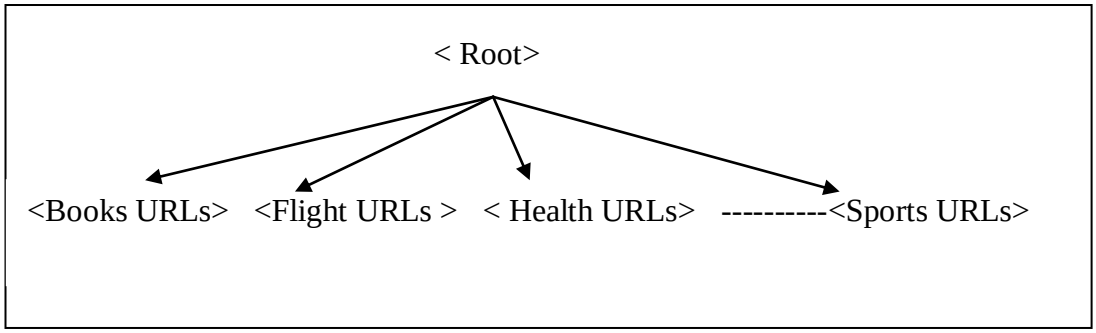


Figure 6.5(b): Domain Matching

Step 3- Delete the keyword that helped to identify the domain from the Query.

$Q' = \{K_1, K_2, K_3, K_4, --, --, --, --, K_n\} - \{K_i\}$

$Q' = \{$Flight Delhi Lucknow$\}-\{$Flight$\}$

$Q = \{$Delhi Lucknow$\}$

Reduce the Query: Delhi Lucknow

In the internal process each domain is assigned a Domain ID as shown in Figure 6.1. For instance the Domain ID of Flight search domain is D2.

Step 4 (Term Matcher): Term Matcher plays very important role for the automatic identification of the semantic relationships between attributes of different search interfaces and also is responsible for searching the query term in the domain specific Index. In Figure 6.6 (attribute representation) which are helpful for term matching like

Fuzzy Matching, Domain Specific Thesaurus, Data Type Matching [15,62,70] and Boolean method.

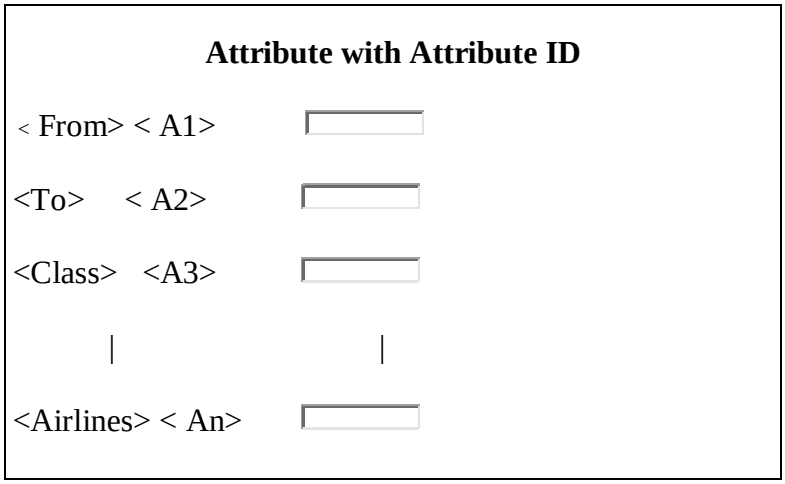


Figure 6.6: Attribute Representation

The hierarchical representation shown in Figure 6.7

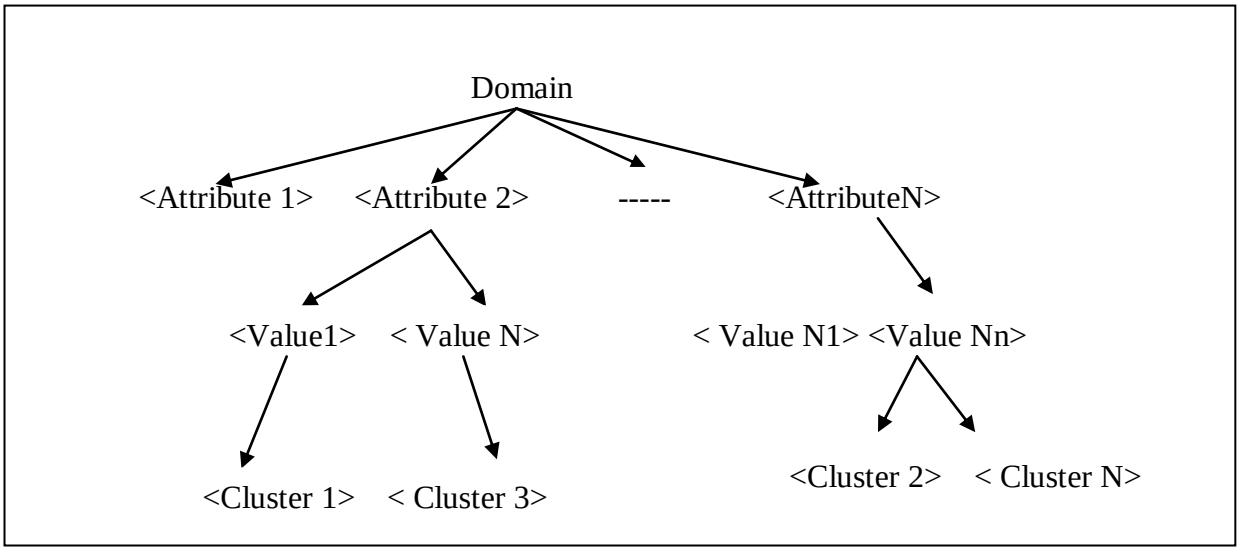


Figure 6.7: Hierarchical Representation of Structure

Reduced Query: Search the query (Q: Delhi Lucknow) in the Schema-Instance based Index.

The various phases are required to search the query term in Schema-Instance based Index.

Phase 1: Each Attribute is assigned an Attribute ID.

Phase 2 -Assign a value from value set to each Attribute.

Phase 3- Submit the form

Phase 4 -The downloaded document corresponding to the form is stored in the page repository.

Step 5: Table 6.3 shown the result page of fired query: Flight from Delhi to Lucknow

Table 6.3 Result Page "Delhi to Lucknow"

S.No.	**From**	**To**	**Airline**	**Time**
1	Delhi	Lucknow	GoAir	6:00am
2	Delhi	Lucknow	Spice jet	9:00 am
3	Delhi	Lucknow	GoAir	1:00pm
----	Delhi	Lucknow	AirIndia	3:00 pm
83	Delhi	Lucknow	GoAir	7:00pm

The next chapter discusses the proposed Ranking Mechanism for Domain Specific Hidden Web in details.

Chapter VII

RANKING MECHANISM FOR DOMAIN SPECIFIC HIDDEN WEB

7.1 INTRODUCTION

The mechanism for processing the query has been discussed in chapter 6. The query processing is based upon the Schema-Instance based Indexing mechanism for the matching the query term, apart from these algorithms for the query processing has been proposed. The advantages of this algorithm are as follows

1. Application of linguistic module to reduce searching time.
2. Identifies the domain from the query.
3. The query is searched on the principle of Schema-Instance based Indexing and provides better quality of result in comparison of keyword indexing because of data stored in structured manner instead of unstructured. Also in this case the redundancy is low due to the Schema-Instance (Attributed-Value pair) based indexing technique is used.

The objective of ranking mechanism is to arrange the results according to their relevance.There are several ranking techniques for efficient ranking of web page has been proposed in literature survey that are compared in Table 7.1. Each technique has its own advantages and disadvantages.

Table 7.1 Comparison Chart of Ranking Techniques

S.No	**Ranking Techniques**	**Technology**	**Score Calculated**
1	Page Rank	Link based	Through Web content- and link-analysis techniques
2	Entity Level Ranking Method	Entity based	Popularity of web page frequency of the entity
3	Probability	Links & Weight	Ranked document by their

	Ranking Principle		approximate probability
4	Weight Rank	Hits & Weight	Hits is counted and count weights increase linearly
5	TF-IDF	DF, IDF	Score

The next section discusses the proposed work flow diagram of ranking and proposed ranking distance method in detail.

7.2 PROPOSED WORK FLOW DIAGRAM OF RANKING MECHANISM

As per the proposed architecture of Least Cost Vertical Search Engine based on DSHWC (Figure 3.1) and proposed work flow diagram of ranking mechanism Figure7.1, when the user issues a query through query interface, the end users are able to convey their information.After receiving of a query from the query interface, the query processor processes the query and generates sub queries for further processing, it is passed through the domain term for quick identification of the domain. The query generated by user is sent to the search engine for processing and then the result is sent back according to the ranking of the web page.

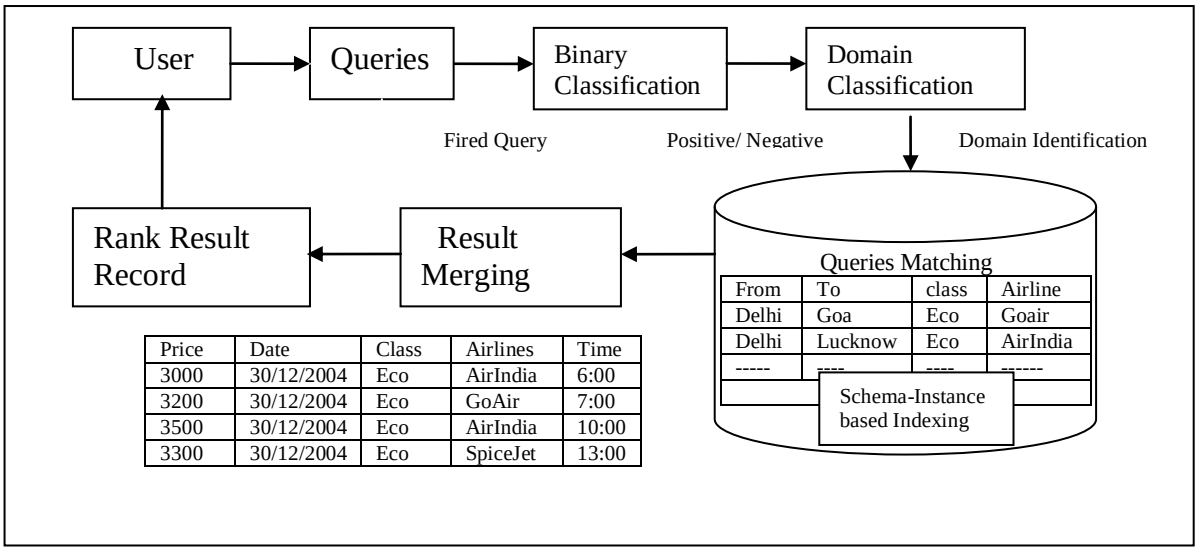


Figure 7.1 Proposed Work Flow Diagram of Ranking

The basic function of binary classification is categorized the query into two categories in form of true or false, positive or negative.After the identification of the domain, term matcher plays very important role for the automatic identification of the semantic

relationships between attributes of different search interfaces. Thereafter, it combines the data with different sources and provides the combined view of the data to the user.

7.3 PROPOSED *RANKING DISTANCE METHOD*

The Ranking Distance (RD) can be defined as the sum of the number of pair wise tuple transactions in database. The Ranking Distance (RD) is measured by the distance between the two tuple records. The pair of two tuple position is represented in the form of Ti and T_j and N is the total number of result records *shown in Table7.2.*

- T_i = i^{th} position of tuple, The starting position of T_i is 2, increment in each step by one (T_{i+1}) and reached at $T_{j=}$ N;
- T_j = j^{th} position of tuple, The starting position of T_j is 1, increment in each step by one (T_{j+1}) and reached at T_i< N;
- N = Total number of tuple in table

Table 7.2: Books Domain (Tuples)

Tuple No.	**Author Name**	**Title**	**ISBN**	**Publisher**	**Subject**	**Keyword**
T_j =1	Charles	Introduction to Algorithm	8120340078	PHI	Computer Science	Algorithm sorting
$T_{i=}$ 2 T_{j+1}	Andy Tanenbaum	Computer Network	9332518742	Pearson	Computer Science	Physical data link access control
T_{i+1}	Andy Tanenbaum	Modern Operating system	8120339045	PHI	Computer Science	Process threads memory
	Willam	Operating System	9332518807	PHI	Computer Science	Memory management
	Siberschatz	Operating System Concepts	8126520515	Wiley	Computer Science	Deadlock system
T_{j+1}	Andrew	Modern compiler Implementation in C/ Java	8131720470	Cambridge University Press	Computer Science	Scheduling Optimization
T_{i+1}	Hopcroft	Introduction to Automata Theory	8131720470	Pearson	Computer Science	Computational complication
N=20	Mike	The complete book of corvette	0760345740	Motor Book	History	Sports car

The user submits a query through the query interface and it passes through the different functional components. The ranking module arranges the result pages according to their relevance, ranks them according to score. The output of this module is an ordered record of Web pages such that the pages on the top of the list are having the highest rank. MT-IMT (Matched Term & Inverse Matched Term) is commonly used method to decrease the number of measurements of document.

A term (query Term) in a record is directly proportional to the frequency of the term and it is referred as matched term (MT). Every term which is used in record assigned a weight. It is fully dependent upon the number of times a particular term become visible in a record. The Inverse Matched Term is used to find frequency of the term in a record. It is inversely proportional to the number of the record (N). Here, N is the number of records. In the first step, a collection of all words used in the tokenized records is generated. In second step, the number of Matched Term (MT) of each word is calculated. In third step, the Inverse Matched Term (IMT) for each word is calculated. Matched Term (MT) for a word is the number of record in which the words occur.

- Where N= Total number of records
- MT= Matched Term
- QT= Query Term
- Score= Query Term and Inverse Matched Term
- RD= Ranking Distance
- RT= Relationship of Records

The formula which is used to calculate IMT is given below.

$$IMT = \frac{N}{MT} - - - - - - - eq. 7.1$$

The formula which is used to calculate score is given below.

$$Score = QT\ Log \frac{M}{MT} - - - - eq. 7.2$$

The formula to compute ranking distance given below:

$$RD = \sum | Ti - Tj | N - - - - \text{eq. } 7.3$$

The purpose of this measurement is to compute the relationship of the given tuple or record with score value and sum of the number of pair wise transaction to find out better result. The relationship of tuples (RT) can be defined as Score of the Query Term (QT)-Inverse Matched Term (IMT) and Ranking distance (RD) to identify the frequency of query term and number of tuples available in the database.

$$RT = Score + RD - - - - - eq. 7.4$$

The frequency calculator determines how many times a particular query is repeatedly issued by different users in the given span of time; it shows the popularity of the query.

7.3.1 Proposed Rank Calculation

When the user searches a book from the Books domain, he/she needs to fill the form. For Books domain, the category of the subject is filled by the user like subject "computer science", it gives the result in tabular or structured format. Here, the total seven outputs appear as the results that are shown in Table 7.3.

Table 7.3: Books Domain: "Subject" Computer Science

S.No.	Author Name	Title	ISBN	Publisher	Subject	Keyword
1	Charles	Introduction to Algorithm	8120340078	PHI	Computer Science	Algorithm sorting
2	Andy Tanenbaum	Computer Network	9332518742	Pearson	Computer Science	Physical data link access control
3	Andy Tanenbaum	Modern Operating system	8120339045	PHI	Computer Science	Process threads memory
4	Willam	Operating System	9332518807	PHI	Computer Science	Memory management
5	Siberschatz	Operating System Concepts	8126520515	Wiley	Computer Science	Deadlock system
6	Andrew	Modern compiler	8131720470	Cambridge University	Computer Science	Scheduling Optimization

		Implementation in C/ Java		Press		
7	Hopcroft	Introduction to Automata Theory	8131720470	Pearson	Computer Science	Computational complication

Thereafter, it identifies the search term in the document to check whether it is available or not. If it is available in document the value assigned is one, if not then assigned the value is zero. The Table 7.4 below contains the terms like subject "computer science", publisher "PHI", Title Operating System" and Author "Andy Tanenbaum". If the user enters the search term by "subject", the outcome of the same would be reflected in the form of a table depicting the availability in all the tuples like Rec1, Rec2, ..., Rec7. Here, "1" represents that the search item is available and "0" represents the non-availability of the search item.

Table 7.4: Assigned Value (Book Domain)

S.No	**Term**	**Search**	**Rec 1**	**Rec 2**	**Rec 3**	**Rec 4**	**Rec 5**	**Rec 6**	**Rec 7**
1	Subject	Computer Science	1	1	1	1	1	1	1
2	Publisher	PHI	1	0	1	1	0	0	0
3	Title	Operating system	0	0	1	1	1	0	0
4	Author	Andy Tanenbaum	0	1	1	0	0	0	0

The proposed Rank Calculation methods apply on the Books Domain to find the Score and distance between the pair-wise records on the result page.

I. Fired Query: Books on subject "Computer Science".

Where QT, MT, IMT is the main source to find the Score.

N total number of records available in database $= 20$

$\mathrm{QT(Fired\ Query\ Term)} = 1$

MT (Matched Term in database) =7

Apply the proposed equation 7.2 to find out the Score.

$$Score = QT\ Log\frac{N}{MT}$$

$$1\ \text{Log}\frac{20}{7} = \text{Log}\ 2.857 = 0.456$$

Apply the proposed equation 7.3 to find out the distance between the pair wise records on the result page of the subject "Computer Science"

$$RD = \sum |\ Ti - Tj\ |\ N$$

$$RD = \frac{\{(2-1)+(3-2)+(4-3)+(5-4)+(6-5)+(7-6)\}}{7}$$

$$RD = \frac{\{1+1+1+1+1+1\}}{7} = \frac{6}{7} = 0.85$$

Apply the proposed equation 7.4 to compute the relationship of specified tuples for better ranking result.

$$RT = Score + RD - - - - - 7.4$$

$$RT = 0.456 + 0.85 = 1.30$$

II. Fired Query: Books on Computer Science by "PHI" Publisher

Where QT, MT, IMT is the main source to find the Score.

N total number of records available in database $= 20$

QT(Fired Query Term) $= 1$

MT (Matched Term in database) =3

Apply the proposed equation 7.2 to find out the Score.

$$1\ \text{Log}\frac{20}{3} = \text{Log}\ 6.67 = 0.824$$

Apply the proposed equation 7.3 to find out the distance between the pair wise records on the result page of the Publisher "PHI"

$$RD = \frac{\{(3-1)+(4-3)\}}{7} = \frac{\{2+1\}}{7} = \frac{3}{7} = 0.428$$

$$RT = 0.824 + 0.428 = 1.252$$

Similarly, the rest search terms are evaluated like: Title & Author. The Value of the Title & Author as shown in the Table 7.5.

Table 7.5: Score Result of Books Domain

Term	**Search**	**Number of Records(N)**	**MT**	**Score**	**RD = ∑ | T_i-T_j | ⁄N**	**R_T = Score + RD**
Subject	Computer Science	20	7	0.456	0.85	=0.456+0.85=1.30
Publisher	PHI	20	3	0.824	0.428	=0.824+0.428=1.252
Title	Operating System	20	3	0.824	0.29	=.824+0.29=1.114
Author	Andy Tanenbaum	20	2	1.0	0.14	=1.0+0.14=1.14

After the calculation of the score the highest value obtained is 1.0 for the term "Author", similarly the second value is .824 for the "Publisher" and "Title" and lowest value is 0.456 for the "computer science". Adding the ranking distance values pair-wise (RD) with score the search term "computer Science" points at the highest value. This show the results available in the database, however, in the database the search term with the highest number of records would possess the highest priority in the ranking for instance the term "computer science" had highest number of records and thus contains the greatest priority of all the other search terms given in the Table 7.6. It also shows the difference between the same values of the score but the pair-wise difference of the record is slightly different, because of the distance of records placed are not same. The final output is shown in the below Table 7.6.

Table 7.6: Final Result of Books domain

Term	**Search**	**Number of Records (N)**	**MT**	**Score**
Subject	Computer Science	20	7	0.456
Publisher	PHI	20	3	0.824
Title	Operating System	20	3	0.824
Author	Andy Tanenbaum	20	2	1.0

When the user searches for a flight from the flight domain, he/she needs to fill the form. Once the form is filled by the user entering the search terms "FROM " –"TO" , the output in the form of basic information like travel date, No of passenger is obtained in a tabular or structured format. When the user fires the query form "Delhi" TO "Lucknow", the total output is generated from the database contains 83 records belonging to the Delhi to Lucknow. The Figure 7.2 showed the values of flight domain from Delhi to Lucknow. To produce a composite weight for each term in each document, the terms with various frequencies in the total collection of 83 documents are presented below in the Table 7.7. For example, the user searches the flight from Delhi to Lucknow a total of 83 flights results appear on the summary screen as shown in the Figure 7.2 below.

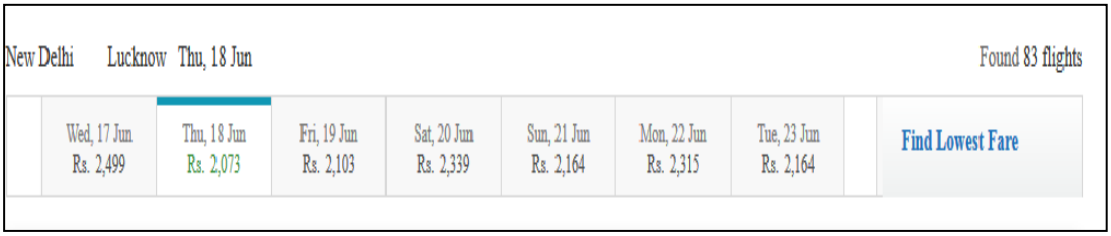


Figure7.2: Flight Search Result (New Delhi to Lucknow)

The Figure 7.2 converted into tabular format for apply the proposed ranking distance method on Flight domain.

Table 7.7 Flight Database "Delhi – Lucknow"

S.No.	From	To	Flight	Time
1	Delhi	Lucknow	GoAir	6:00am
2	Delhi	Lucknow	Spice jet	9:00 am
3	Delhi	Lucknow	GoAir	1:00pm
----	Delhi	Lucknow	AirIndia	3:00 pm
-----	Delhi	Lucknow	AirIndia	5:00 pm
----	Delhi	Lucknow	AirIndia	6:00 pm
83	Delhi	Lucknow	GoAir	7:00pm

II. Fired Query: Flight from New Delhi to Lucknow

Where N

$=$ total number of search records shown in Figure 7.6

QT(Fired Query Term) $= 1$, MT (Matched Term in database) =83

Apply the proposed equation 7.2 to find out the Score.

$$Score = QT\ Log\frac{N}{MT} - - - -7.2$$

$$1\ \text{Log}\frac{83}{83} = \text{Log}\ 1 = 0$$

Apply the proposed equation (7.3) to find out the distance between the pair wise records on the result page.

$$RD = \sum |\ Ti - Tj\ |\ N - - - -7.3$$

$$RD = \frac{\{(2-1)+(3-2)+(4-1)+ - - +(83-82)\}}{83} = \frac{82}{83} = 0.99$$

Apply the proposed equation (7.4) to compute the relationship of specified tuples for better ranking result.

$$RT = Score + RD$$

$$RT = 0 + 0.99 = 0.99$$

Thereafter, it identifies the term which is available in record or not. If it is available in record then value is assigned one, if it is not available the assigned value is zero. The Table 7.8 contains the flight search FROM “Delhi” TO “Lucknow”.

Table 7.8: Assigned Value (Flight Domain)

S.No	**FROM**	**TO**	**Record 1**	**Rec 2**	**Rec 3**	**Rec----**	**Rec 83**
1	Delhi	Lucknow	1	1	1	1	1
2	Delhi	Lucknow	1	1	1	1	1
3	Delhi	Lucknow	1	1	1	1	1
--	Delhi	Lucknow	1	1	1	1	1
83	Delhi	Lucknow	1	1	1	1	1

The score value is 0 for flight “All “search FROM “Delhi” to “Lucknow”. After the addition of the ranking distance values pair-wise RD with score of the flight search, the final value reaches at 0.99. This shows the value Flight “Go Air” having three records in the position of 1st, 3rd and 83rd. It also shows the difference between the same values of the score but the pair-wise difference of the record is slightly different, because of the distance of records placed are not same.

Table 7.9: Score Result of Flight Domain

Term	Search	Number of Records	MT	Score
All	Delhi Lucknow	83	83	0
GoAir	Delhi Lucknow	83	3	1.44

7.3.2 Result Merging

The results of local queries must be interpreted and assembled according to the global join conditions. Logically equivalent data items may be implemented differently in terms of format, scale, and encoding in different systems. The all query schema interface integration conceptually consists of just one relation schema for domain. The global relation consist set of tuples, set of attributes and set of real word entities. The Relation R1, R2, ….., Rn are to be integrated for a relation S. The QI does not allow the same attributes to appear multiple times, for airline domain used three relations {Option, Status and Passenger} and one materialized view for improve query performance in term of access time, maintenance cost [70].

7.4 OUTCOME OF PROPOSED WORK

The comparison chart between the proposed work and the existing techniques is shown in the Table 7.10 (I & II) for Books domain & Flight domain. According to the comparison Table 7.10 (I & II), it can be observed that the search terms “computer science” and “Go Air” were assigned the lowest value of priority in the conventional system of search. However, the proposed system rightly assigns the highest value of the priority to the search term on the basis of the fields discussed in the previous sections.

Table 7.10 (a) Comparison Chart Book Domain

Score (QT-IMT)					Proposed work	
Term	**Search**	**Number of Records(N)**	**MT**	**Score**	**RD = ∑ | T_i-T_j | ∕N**	**RT = Score + RD**
Subject	Computer Science	20	7	0.456	0.85	=0.456+0.85=1.30
Publisher	PHI	20	3	0.824	0.428	=0.824+0.428=1.252
Title	Operating System	20	3	0.824	0.29	=.824+0.29=1.114
Author	Andy Tanenbaum	20	2	1.0	0.14	=1.0+0.14=1.14

Table 7.10 (b) Comparison Chart Flight Domain

Flight	**Fro**	**TO**	**Number of Records(N)**	**MT**	**Score**	**RD = ∑ | T_i-T_j | ∕N**	**R_T=Score+ RD**
All	Delhi	Lucknow	83	83	0	0.99	0.99
Go Air	Delhi	Lucknow	83	3	1.44	0.99	2.43

As per the proposed model the final result Table 7.11 given below displays the ranks allocated to all the search terms generated by the user according to the ranking distance method. So it can be inferred from the specified ranking method, that the proposed scheme allocate the highest priority to the user search as compared to the conventional search engine. Also the proposed algorithm is designed in such a way that lead to reduction in the search query time.

Finally, the proposed ranking method also provides more appropriate and useful data to the users thereby increasing the efficiency.

Table 7.11: Proposed Ranking Rank Result

S.No	Possible Word Search	Result	Ranks
Domain Book			
1	Computer Science	1.30	1st
2	PHI	1.25	2nd
3	Operating System	1.14	3rd
4	Andy Tanenbaum	1.11	4th
Flight Domain			
5	Go Air	2.43	1st
6	All	0.99	2nd

The next chapter discusses the Predicting the Next Query for Domain Specific Hidden Web in detail.

Chapter VIII

PREDICTING THE NEXT QUERY DOMAIN SPECIFIC HIDDEN WEB

8.1 INTRODUCTION

The working detail of the proposed ranking mechanism has been discussed in the previous chapter. The flow diagram of ranking and rank calculation on the basic of (QT-IMT) Score and proposed Ranking Distance (RD) method to find better ranking result was presented. With the drastic increase of web information, getting the desired information has become difficult and is on the rise day by day with increasing internet users. In order to provide the useful information to the user, the search pattern of the user should be saved and prediction on the basis of the analysis of the user search pattern should be made.This type of approach is usually termed as the user centric approach.

The user generates the request to the web server for searching the domain specific search for example, if a user wants to search information about Flight domain, then in order to get the required information, he/she must go to the particular site and fill the details in a search form available on the site. As a result he/she gets the desired information. Various research papers [98,99,100,101,102,103,104,105,106,107] point out the different methods of query prediction like NLP, data mining, web mining techniques. For the better query prediction two type of logs are required they are the Query log and the Web Server Log as shown in the Figure 8.1.

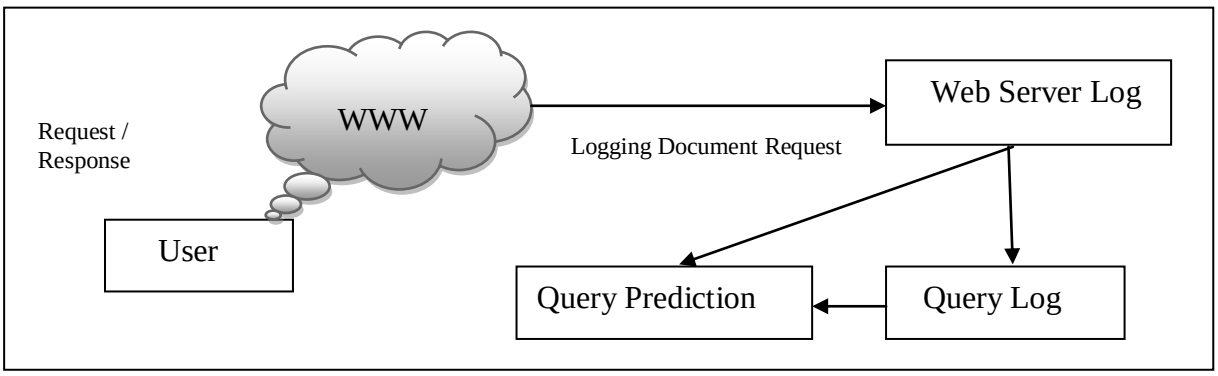


Figure 8.1: Web Server Systems

The next section discusses the proposed architecture of next query prediction in detail.

8.2 PROPOSED ARCHITECTURE OF NEXT QUERY PREDICTION

Resources are responsible for all the documents either approaching directly from the Web or some other network source. Network profile contains the network capabilities description**.** The major parts of proposed work are the Data Cleaning, Visited Paths, Transactions and Query Prediction. The proposed Next query prediction architecture is shown in Figure 8.2. It s employs by seven components. The major components of the proposed work as follows:

1. Search Interface
2. Web Log
3. Data Cleaning
4. Query Log
5. Path Analysis
6. Association Rule
7. Query Prediction

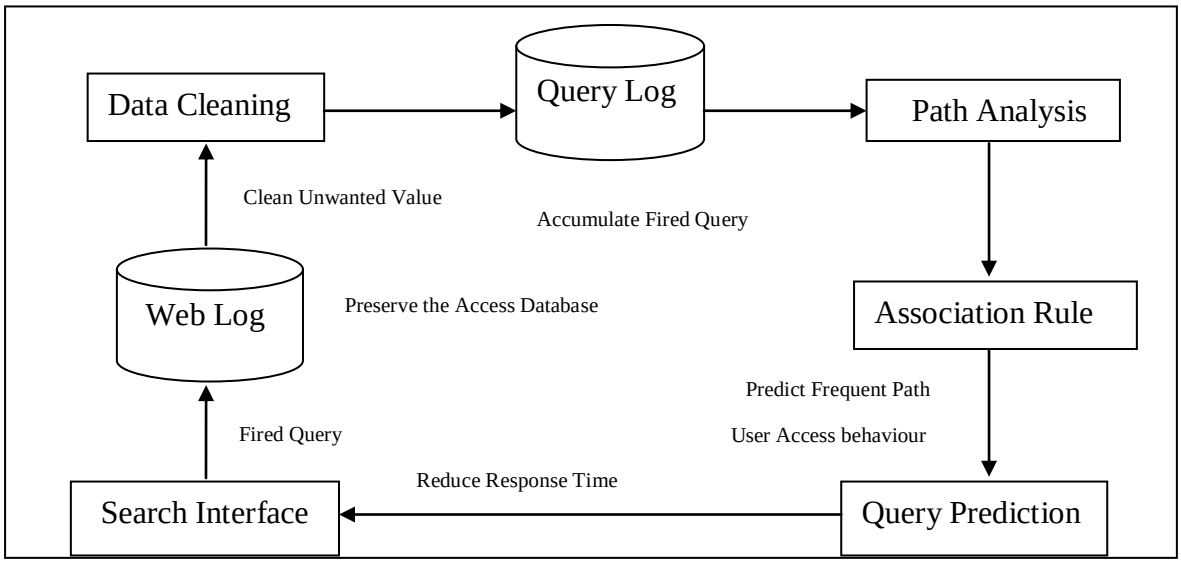


Figure 8.2: Proposed Architecture for Next Query Prediction

8.2.1 Search Interface

The proposed architecture refers to the user who generates the query and provides access to the system. For example, user searches for the flight ticket for a certain destination. The interface is shown in Figure 8.3.

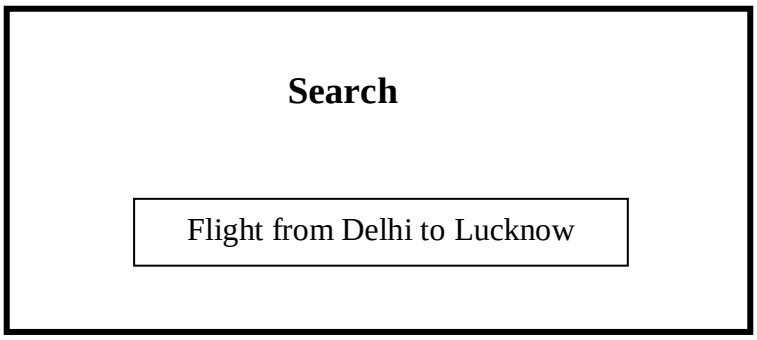


Figure 8.3: Search Interface

8.2.2 Web Log

Once the form is filled by the user, the user wants to search flight to any destination. The Weblog maintains the database of the queries accessed by the user [73]. Here, client IP address plays very important role for the next query prediction. The search term fired by the user is stored in database as shown in the Table 8.1. This table is referred to as Web Log table. The Web Log maintains the list of actions performed by the user. Whenever the user accesses the Domain Specific Web Search, the Web Log updates the information and frequently stores it. Further every Web server maintains the list of actions performed/requested by the user into Web Log files. The architecture for Web access processing is shown in the Figure 8.2. According to the Figure 8.2 the client file consisting of the user query is stimulated into the presentation and interaction. These are employed to provide the structured information. The different tiers followed in access log processing are client, presentation & interaction, retrieval & analysis, and resources. First the data entered is cleaned, second the cleaned data is sent for the transaction layer. It is also responsible for information presentation and user interaction. The users access the Web pages according to the users preferences.

Table 8.1 and Table 8.2 show the required information in the form of columns which are important and maintained in the Web Log. Email Id, Access Time are not mandatory but are optional, however they are required for the authentication of the user. Web Log files are converted into relation schema with set of attributes set of tuples or set of real words entities. Data processing is used to process the data so as to make relational schema useful for getting information.

- Client IP Address: It is represented by IP address, for example (10.184.0.7)
- Data Reduction: Remove undesired fields, avoiding some attributes which do not directly influence the results, reduces data set. Hence, from current log data, attributes User ID, User Action, and File Path are omitted.
- Only successful requests are filtered.

Table 8.1: Web Log (Flight Domain)

Client IP address	**Email ID**	**Login count**	**Access Time**	**Time Interval**	**Trip Option**	**HTTP protocol**	**Status code**	**File size(KB)**
10.184.0.7	aa@gmail.com	1	3:34:11 /5/april/2013	10	One way	1.0	200	2048
10.184.0.18	su@hotmail.com	1	4:23:11 /5/april/2013	15	Round Trip	1.0	200	2056
10.184.0.20	ms@yahoo.com	1	4:23:11 /5/april/2013	20	Multicity	1.0	200	2056
10.184.0.21	kb@yahoo.com	1	5:24:11 /5/april/2013	20	Package	1.0	200	2056
10.184.0.27	tr@gmail.com	1	5:25:12 /5/april/2013	30	One Way	1.0	200	2056

- If status code=200 request successful; record is retained.

  If status code! =200 request unsuccessful; record is removed.

Frequency calculator determines how many times many times a particular query is repeatedly issued by different user

Table 8.2: Web Log with Query

Client IP Address	**Query**	**Query Frequency count**
10.184.0.7	Flight from Delhi to Lucknow	1
10.184.0.18	Flight from Goa to Mumbai	1
10.184.0.20	Flight from Delhi to Goa	1
10.184.0.21	Flight from Goa to Lucknow	1
10.184.0.27	Flight fron Delhi to Pune	1

25.180.0.23	Books on Computer Science	1
28.185.0.6	YMCA University	1

8.2.3 Data Cleaning

Data Cleaning is a process to clean the unwanted symbol, punctuation, removed the common and most frequent keywords from the data shown in Table 8.3. For example, a user fires a query to search any Flight from Delhi to Lucknow, before processing the transactions unwanted value and symbol are omitted.

Query: Flight, from, Delhi, to Lucknow.

Step1: Apply Linguistic modules such as Tokenization, Stop-word removal, Lemmatization and Normalization before processing the query .After applying the tokenization approaches the out of the query look like:

Output: Flight from Delhi to Lucknow

Step 2: Query Input: Flight from Delhi to Lucknow: Which contains two stop words (from, to) in the removing process the system check the list of the stop-words. Once the verification process is over from the query, all the stop-words are removed. After the removing stop words: Flight Delhi Mumbai

Table 8.3: Clean Query Data

Client IP address	**Access Time**	**Time interval**	**Query**
10.184.0.7	3:34:11/5/april/2013	10	Delhi Lucknow
10.184.0.18	4:23:11/5/april/2013	15	Goa Mumbai
10.184.0.20	4:23:11/5/april/2013	20	Delhi Goa
10.184.0.21	5:24:11/5/april/2013	20	Goa Lucknow
10.184.0.27	5:25:12/5/april/2013	30	Delhi Pune
25.180.0.23	8:27:13/5/april/2013	1	Computer Science

Step 3- Delete the keyword that helped to identify the domain from the Query

Final Output = {Delhi Lucknow}

After the cleaning, correcting and processing the data web log file maintains the set of attributes like client IP address, email ID, Login Count, Access Time, Trip Option, HTTP protocol, Status Code, File Size, etc.

8.2.4 Query Log

Query Log contains the record of queries fired by different users at different time shown in Figure 8.4. It stores the database on the basis of the keywords of various domains and helps in the searching or firing the query.

Query Log List

- Q1: Flight Delhi to Mumbai
- Q2: University YMCA
- Q3: Income tax
- Q4: Network protocol
- Q5: Medical health checkups
- Q6: Book on cloud computing
- Q7: 2BHK Flat
- Qn: Cricket News

Figure 8.4: Query Log

Thereafter, the history of user access behaviour is stored in Query Log as well as in Web Log shown in Table 8.4. However the Query Log only contains the fired query information. The Web Log stores each and every information like IP address, E-mail ID, HTTP information, status, files size, time duration of the user accessing etc.

Table 8.4: Query Log (Flight Domain)

Client IP address	From	To	Trip Option
10.184.0.7	Delhi	Lucknow	One way
10.184.0.18	Goa	Mumbai	Round Trip

10.184.0.20	Delhi	Goa	Multicity
10.184.0.21	Goa	Lucknow	Package
10.184.0.27	Delhi	Pune	One Way

8.2.5 Path Analysis

Path Analysis is responsible for the analysis and retrieval of all the documents [74]. Further the analysis is divided into two operations mainly categorized as the path analysis and association rule. The Path Analysis of user access depends upon the forward traversal path and backward traversal path. Path achievement depends upon the forward traversal path and backward traversal path. Flight search" form" yatra.com home page shown in Figure 8.5. Once the form filled by the user, search engine provides link of the next form/page i,e called forward reference. Forward traversal path is sequence of the connected pages in a web presentation where no page is previously visited. The connected traversal path can be described as follows (- → Home page), (Homepage→Option Check), (Option Check→ Booking Page), (Booking Page→Availability Check),(Availability Check → Make Payment). If user clicks or press close or back button then backward reference occurs.

For example: query fired by the user with IP address for example, 10.184.0.7

From: Delhi

To: Lucknow

Trip Option: One Way

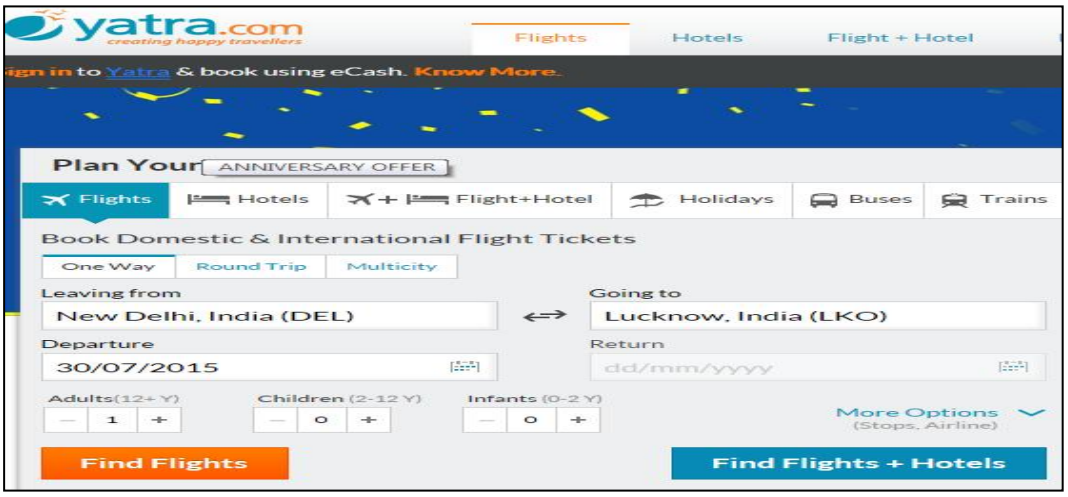


Figure 8.5: Homepage (Yatra.com)

Case 1: Access path is Home Page- Option Check Page. There is direct path from Home Page to Flight detail page shown in Figure 8.5

Case 2: If user clicks on Book Now Button then forward reference occur then move on availability page. (Homepage-Option Check Page- Availability Page) Shown in Figure 8.5

Case 3: If user clicks on Availability Page – Make Payment then again forward reference occur. The sequence is appear like (Homepage-Option Check Page- Availability Page- Make Payment Page)

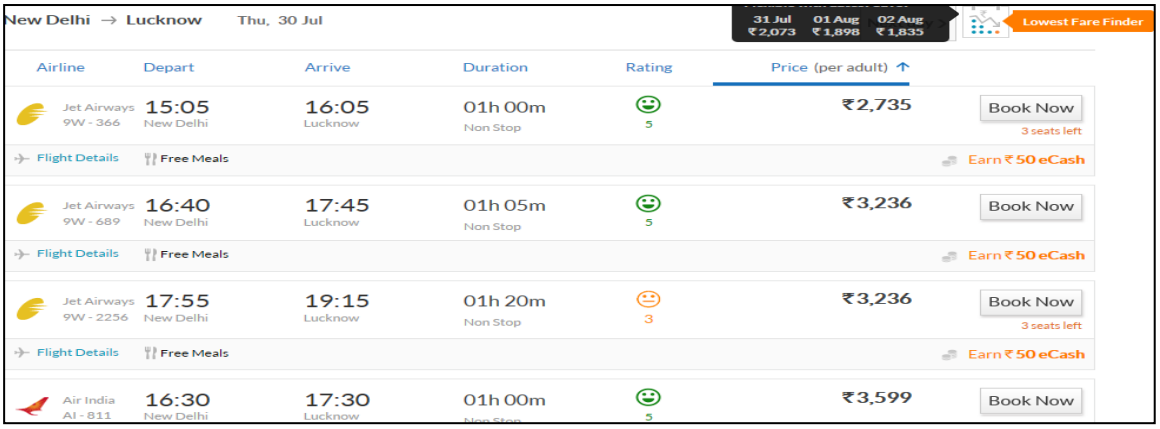


Figure 8.6: Option Page (Yatra.com)

Case 4: If user clicks on back/close the page then backward reference occurs.

8.2.6 Association Rule

Association Rule is used to extract and predict frequent path and user access behaviour according to the rules [74].

- Total number of transactions N=5
- Let's assume that minimum support level = 15% =0.75=1
- Minimum confidence level = 50 %

Because the minimum support is 1, the Table 8.5 shows frequent 1- 2 item set, support for trip option and subsequent analysis. The frequent trip options are {One Way}, {Round Trip}, {Multicity}, {Package}. The most frequent trip option is {One Way}

- *Rule 1: $\forall x \in$transaction, contain (X, One Way)=>contains (X, Package)*
- Confidence= support (One Way, Package) /support (One Way) = 2/2 =100%

Rule Established

- *Rule 2: $\forall x \in$transaction, contain (X, Package) =>contains (X, One Way)*
- Confidence= support (Package, One way)/support (One Way) = 1/2 =50%

Rule Established. This rule is strong rule. The users navigating package also are visiting one way trip.

- *Rule 3: $\forall x \in$transaction, contain (X, Multicity) =>contains(X, One Way)*
- Confidence= support (Multicity, One way)/support (One Way) = 1/2 =50%

Rule Established. This rule is strong rule. The users navigating Multi city also are visiting one way trip.

- *Rule 4: $\forall x \in$transaction, contain (X, Round Trip) =>contains (X, One Way)*
- Confidence= support (Round Trip, One Way)/support (One Way) = 1/2 =50%

Rule Established.

Table 8.5: Trip Option Analysis (Flight Domain)

Trip Option	**From**	**To**	**Support Count**	**Support**	**Confidence**
One way	Delhi	Mumbai	2	40%	100%
Round Trip	Delhi	Goa	1	20%	50%
Multi city	Pune	Goa	1	20%	50%
Package	Pune	Delhi	1	20%	50%

It is evident from the Table 8.5 that the one-way trip option chosen by the user has the maximum confidence percentage and also the largest support count of 2. According to the inference of the Table 8.5 it is concluded that the users preferred to buy one-way trip flight tickets as compared to round trip, multi-city and packaged tickets. Similarly, the users going on leisure trips preferred the lowest cost but longer route (with more halts) tickets in comparison to high cost but direct routed flight tickets.

8.2.7 Query Prediction

The working of the Query Prediction shown in Figure 8.7.The Schema-Instance based Indexing used in Query Prediction.

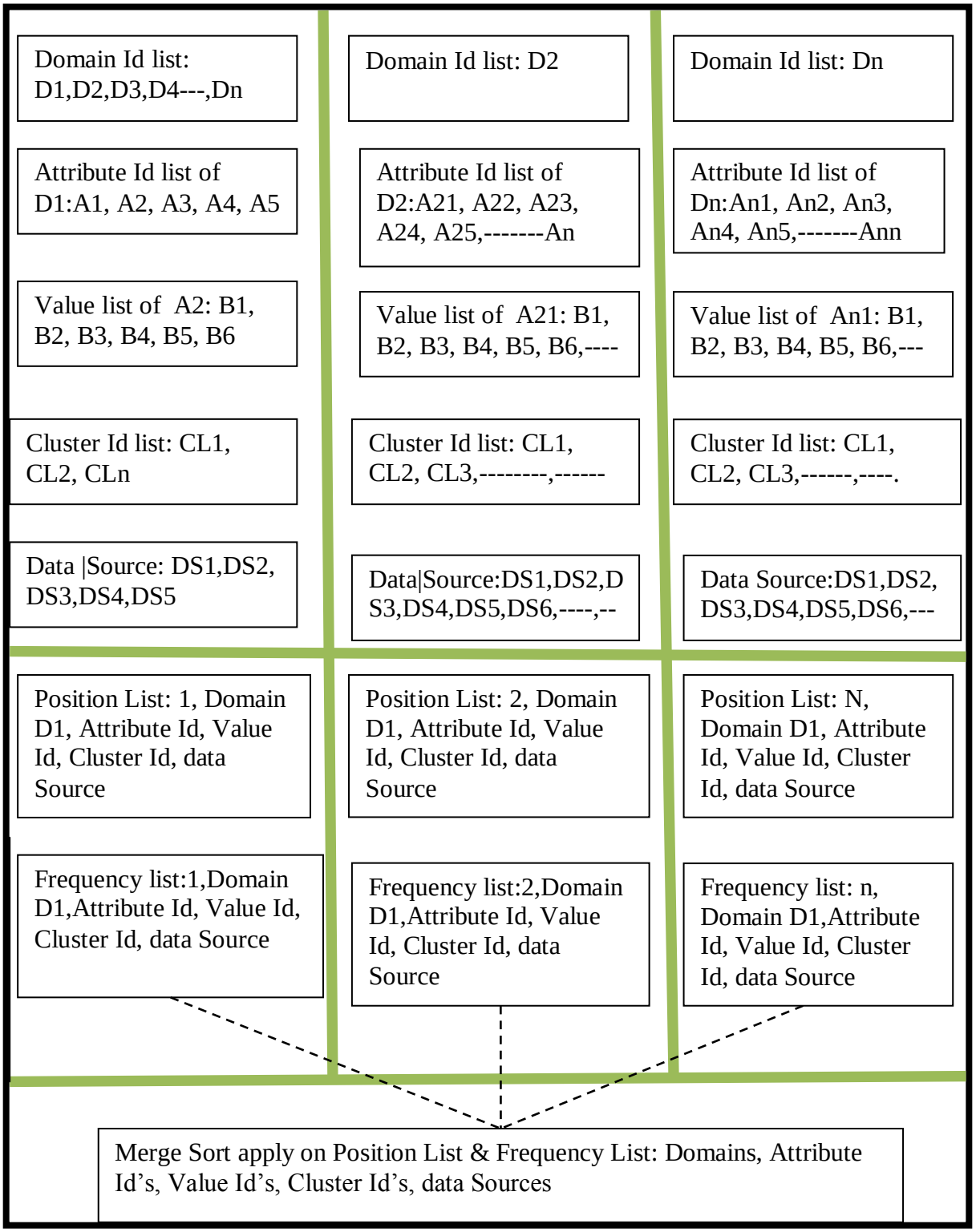


Figure 8.7 Query Prediction Process

The Attributes are taken as the origin of Hidden Web documents for a particular domain.The various domains like Sports, Flight, Education, Health, Books, Mass Media represented as D1,D2,D3,.....,Dn shown in Figure 5.1 in chapter 5, each domain contains the Domain ID, Attributes ID, Value ID, Cluster ID and data source of that particular domain. Thereafter, query processing and ranking method is performed. Through the ranking method position of the domain, attributes, values, cluster and its data sources are decided. Frequency list of domain and its attributes, values, cluster, data source find out through the transaction performed by the users. Transaction process is confirmed after the value stored in Buy Log and Query Log. Association Rule is helpful to extract and predict frequent path and user access activities according to the rules. In the last step apply merge sort technique to find out the frequent domain, attribute accessed by the user.

The next chapter discusses the proposed overall results of the Least Cost Vertical Search Engine based on Domain Specific Hidden Web Crawler (DSHWC).

Chapter IX

IMPLEMENTATION & RESULT ANALYSIS OF LEAST COST (LC) VERTICAL SEARCH ENGINE BASED ON DSHWC

9.1 INTRODUCTION

The Least Cost Vertical Search Engine based on DSHWC for Hidden Web proposed in this thesis provides the key features laying emphasis on saving the search time of the user and thus attain higher cost effectiveness. The searching time is reduced by reducing the response time of crawling, access time is reduced through schema instance indexing mechanism, ranking of the record provides relevant information, effective query processing methods provides fast and accurate results, next query prediction helpful to create history of the user access sites and load balancer monitor / manages the network bandwidth and load. It has been divided into following five phases

- Load sharing architecture of domain specific hidden web
- Indexing of Hidden Web contents for efficient retrieval
- Domain specific query processing
- Ranking mechanism for domain specific hidden web
- Predicting the next query for domain specific hidden web

The details of implementation & performance analysis of each phase have been described in this chapter.

9.2 PERFORMANCE METRICS

Precision (P), Recall (R), Accuracy (A) and F-measure (F) are the parameters for measuring the performance of the Least Cost Vertical Search Engine based on DSHWC. For is the following parameters or performance metrics are taken With the help of above defined terms TP, FP and TN, let us now further define the various performance metrics:

1. *Precision* is defined as a fraction of connected records that are classified correctly and searched by Least Cost Vertical Search Engine based on DSHWC. Mathematically, the *Precision* is given by

$$P = \frac{TP}{TP + FP} — eq. 9.1$$

   - True Positive (TP): The number of correctly searched records.
   - False Positive (FP): The number of non relevant records searched.
   - True Negative (TN): The numbers of hidden web records that are not searched by DSHWC.

2. *Recall* is defined as a fraction of connected records to categories that are classified correctly by Least Cost Vertical Search Engine based on DSHWC. The *Recall* of the proposed system is given by the expression given below

$$R = \frac{TP}{TP + TN} — eq. 9.2$$

3. *Accuracy* is defined as

$$A = \frac{TP + TN}{TP + TN + FP} - eq. 9.3$$

4. *F-measure* is measure of accuracy that considers both precision and recall. *F-measure* is given by

$$\mathrm{F} = \frac{2(\mathrm{Precision})(\mathrm{Recall})}{\mathrm{Precision} + \mathrm{Recall}} - \mathrm{eq.}\ 9.4$$

9.2.1 Data Sets

For experimental evaluation of the proposed work, the following two domains have been considered:

1. Books
2. Airline

A seed URL for each domain was provided to *Search Interface Crawler* and about 200-250 search interfaces were collected for each domain. A sample of two search interfaces pertaining to Books domains are shown from Figure. 9.1 and Figure 9.2

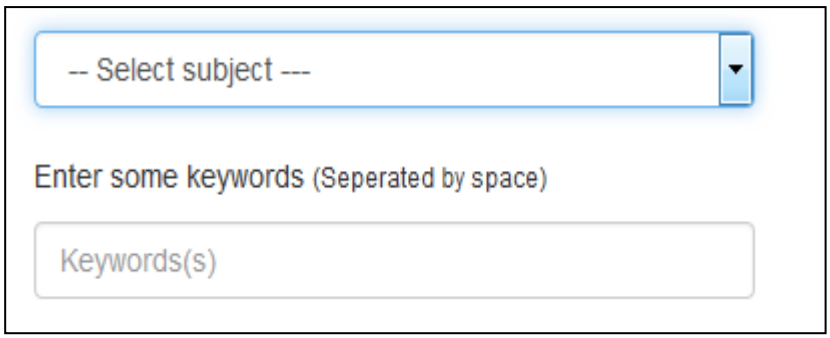


Figure 9.1: Search Interface-1

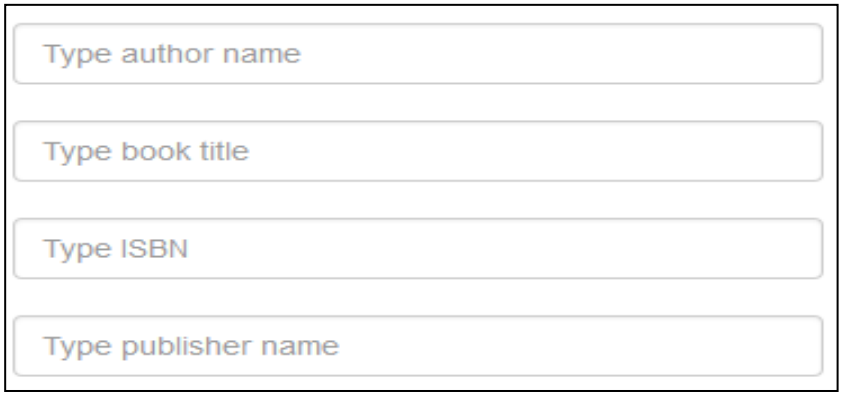


Figure 9.2: Search Interface-2

9.2.2 Experiments

In the beginning the Search interfaces were obtained from different Web sites. In order to obtain a least cost model of vertical search engine based on DSHWC, optimization of response time and full utilization of bandwidth using suitable load balancers, efficient ranking, indexing and through the next query prediction mechanism. The response pages thus obtained were analyzed in order to calculate precision, Recall and F-measure.

A detailed discussion on the performance of every phase of least Cost Vertical Search engine DSHWC is given in the following sections.

9.3 LOAD SHARING ARCHITECURE OF DOMAIN SPECIFIC HIDDEN WEB

The crawling depends upon the size of database and the communication rounds between the crawler and the Web server. If there are 3,00,000 records, then more number of communication/ crawling rounds are required to fetch all the records and

store these obtained records in the page repository. According to the proposed architecture, this is done by load balancer which allows the multiple parallel instances of hidden web crawler to crawl the records simultaneously. Thus, the response time was reduced due to the creation of multiple instances of hidden web crawler which is managed by the load balancer. Also full utilization of network bandwidth was achieved. For instance, if there are small number of records, say 20 records available against a certain fired query, then only one communication round is required to obtain the desired result. The Figure 9.3 shows the server traffic which is transmitted from server to client in terms of data received and sent per hour. Figure 9.3 depicts the total experimental result data sent 1,570 per hour 1.02 k, per minute 16.97 and per second 0.28.

Server traffic: These tables show the network traffic statistics of this MySQL server since its startup.

Traffic [1]		ø per hour
Received	103 KiB	67 KiB
Sent	559 KiB	362 KiB
Total	662 KiB	429 KiB

Connections		ø per hour	%
max. concurrent connections	1	---	---
Failed attempts	0	0.00	0.00%
Aborted	0	0.00	0.00%
Total	138	89.50	100.00%

Query statistics: Since its startup, 1,570 queries have been sent to the server.

Total	ø per hour	ø per minute	ø per second
1,570	1.02 k	16.97	0.28

Query type		ø per hour	%	Query type		ø per hour	%	Query type		ø per hour	%
select	412	267.195	28.77%	show create table	51	33.075	3.56%	show master status	4	2.594	0.28%
set option	324	210.124	22.63%	insert	44	28.535	3.07%	show slave status	4	2.594	0.28%
change db	221	143.326	15.43%	show keys	33	21.402	2.30%	create view	2	1.297	0.14%
stmt prepare	108	70.041	7.54%	show variables	27	17.510	1.89%	show plugins	2	1.297	0.14%
stmt execute	108	70.041	7.54%	show databases	10	6.485	0.70%	show grants	2	1.297	0.14%
stmt close	108	70.041	7.54%	show storage engines	8	5.188	0.56%	show charsets	2	1.297	0.14%
show table status	102	66.150	7.12%	show triggers	8	5.188	0.56%	show collations	2	1.297	0.14%
show fields	80	51.883	5.59%	delete	7	4.540	0.49%	truncate	1	0.649	0.07%
show tables	77	49.937	5.38%	show binlogs	6	3.891	0.42%	show status	1	0.649	0.07%

Figure 9.3: Server Traffic

The size of database in book domain is twenty. This shows that there are twenty records available against the book domain in the proposed implementation prototype.The various attributes used in book domain are specified as bookid, keywords, firstname, lastname, title, ISBN, publisher and subject. Here the bookid is unique attribute in database. Book domain database shown in Figure 9.4

bookId	keywords	firstName	lastName	title	ISBN	publisher	subject
1	friendship unforgettable betrayal	khaled	hosseini	kite runner	1594480001	riverhead trade	literary
2	algorithms sorting quicksort augmenting	charles	e. leiserson	introduction to algorithms	8120340078	phi learning pvt limited	computer science
3	physical data link access control	andy	tanenbaum	computer networks	9332518742	pearson	computer science
4	processes threads memory	andy	tanenbaum	modern operating system	8120339045	phi learning	computer science
5	memory management virtual exclusion	william	stallings	operating system: internals and design principles	9332518807	pearson india	computer science
6	deadlock file system	Silberschatz		operating system concepts	8126520515	WILEY	computer science
7	scheduling optimization compiler design	Andrew	Appel	modern compiler Implementation in c/java	817596071X	CAMBRIDGE UNIVERSITY PRESS	computer science
8	computational complications	Hopcroft	Motwani and Ullman	Introduction to Automata Theory, Languages, and Co...	8131720470	Pearson India	computer science
9	David McCullough, two-time winner of the Pulitzer ...	David	McCullough	The Wright Brothers	1476728747	Simon & Schuster	Aerospace
10	System Dynamics includes the strongest treatment o...	William	Palm III	System Dynamics	0073398063	McGraw-Hill	Aerospace
11	moonshot thinking, and crowd-powered tools to crea...	Peter	H. Diamandis	Bold: How to Go Big, Create Wealth and Impact the ...	1476709564	Simon & Schuster	Civil & Environmental
12	ontainers from Newark to Houston. From that modest...	Marc	Levinson	The Box: How the Shipping Container Made the World...	0691136408	Princeton University Press	Civil & Environmental
13	well written, well paced, fun, and informative. I ...	Charles	Platt	Make: Electronics (Learning by Discovery)	0596153740	Make	Mechanical
14	agined. They will drive themselves, won't consume ...	Levi	Tillemann	The Great Race: The Global Quest for the Car of th...	1476773491	Simon & Schuster	History
15	flagship sports car has become a timeless part of ...	Mike	Mueller	The Complete Book of Corvette - Revised & Updated:...	0760345740	Motorbooks	History
16	omobile transportation to the masses was falling b...	A. J.	Baime	Go Like Hell: Ford, Ferrari, and Their Battle for ...	0547336055	Mariner Books	History
17	Machinery's Handbook has been in continuous public...	Erik	Oberg	Machinery's Handbook, 29th	083112900X	Industrial Press	Mechanical
18	monly referred to as the Pipe Bible, this is the m...	W. V.	Graves	The Pipe Fitters Blue Book	0970832125	Graves Pub Co	Mechanical
19	Discover how to achieve release-quality mixes even...	Mike	Senior	Mixing Secrets for the Small Studio	0240815807	Focal Press	Electrical & Electronics
20	Used worldwide by electricians, engineers contract...	Jones & Bartlett	Learning	Ugly's Electrical References	1449690777	Jones & Bartlett Learning	Electrical & Electronics

Figure 9.4: Book Domain Database

The cost measured in terms of response time. The response time measured

- Time spend in submitting the query at server
- Time spend in retrieving the result
- Receive time from server to Client

The profiling (test speed) status and time of each parameter is shown in Figure 9.5. The searching of the data from the page repository and gives the output on the user screen and passes through different phases of the search engine. To display 20 records of book domain records takes 0.0019s through implemented least cost vertical search engine based on DSHWC. In profiling starting times, the system takes around 0 .000052s, opening tables takes 0.000036s and final output is generated by single Hidden Web Crawler in 0.0019s.

Profiling

Test your speed

Status	Time
starting	0.000052
checking permissions	0.000010
Opening tables	0.000036
System lock	0.000025
init	0.000029
optimizing	0.000009
statistics	0.000019
preparing	0.000012
executing	0.000003
Sending data	0.000187
end	0.000007
query end	0.000003
closing tables	0.000011
freeing items	0.000231
logging slow query	0.000007
cleaning up	0.000006

Showing rows 0 - 19 (20 total, Query took 0.0019 sec)

```
SELECT *
FROM `books`
LIMIT 0 , 30
```

Figure 9.5: Profiling (Test Speed)

The size of the book domain is 15.9 KB. Figure 9.6 shows the three sample record time, such that if 20 records are fetched the query takes 0.0022secs, for 5 records the time taken to retrieve the query is 0.0020s.

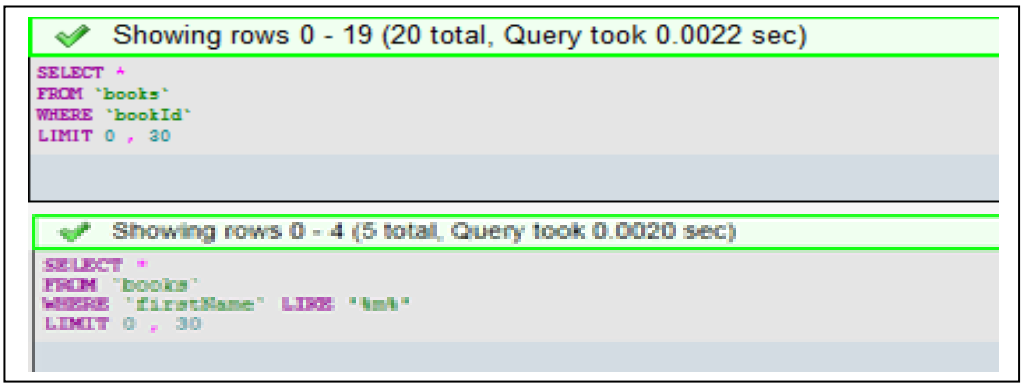


Figure 9.6: Query Time

Figure 9.7 shows the plot depicting the linear relationship between the matched records fetched by the crawler and the response time of the system.

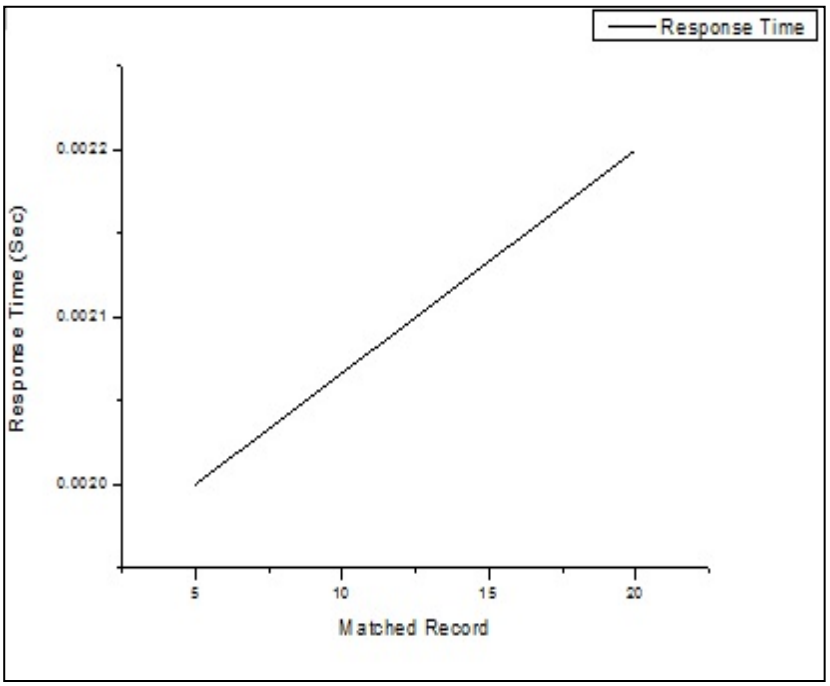


Figure 9.7: Matched record v Response Time

The graph shown in the Figure 9.7 shows the relation between the response time and the number of records. As per the graph the time taken to retrieve the information by single crawler is 0.0019s for 20 records where all values are retrieving with general query, for five records it is 0.0020s for specific results and for 20 records it is 0.0022s

according to the information retrieving by bookid. To increase the efficiency and reliability of the system, the architecture of the proposed least cost vertical search engine based on DSHWC proposed multiple instances of Hidden Web Crawler with dedicated page repository and indexing mechanism employed for obtaining relevant and fast results. It is managed and controlled by the load balancer. The formula for parallel computation is proposed below.

$$Total\ number\ of\ matched\ record = N ---------- eq.4.1$$

$$The\ time\ to\ complete\ by\ single\ crawler = NT ------ eq.4.2$$

$$crawler\ be\ employed = K ---------- --eq.4.3$$

$$Wait\ time\ of\ crawler = Q, Where\ Q\ is\ negligible ---- eq.4.4$$

$$Actual\ response\ time\ by\ single\ crawler = [NT + Q] -- eq.4.5$$

$$The\ expected\ time\ to\ complete\ by\ K = \frac{[NT+Q]}{K} ------ eq.4.6$$

Actual completion time:

I) 0 .0019 sec for 20 matched record (all information)

II) 0.0020 sec for 5 matched record (specific information)

III) 0 .0022 sec for 20 matched record (condition based information)

The actual completion time shown in Table 9.1

Table 9.1 Actual completions Time

No. of Process	Matched Record	Response Time(sec)	Searching Method
1	20	0.0019	All Information
1	5	0.0020	Specific Information
1	20	0.0022	Condition based Information

Figure 9.8 shows the graph comparing the matched records versus response time of different techniques of searching. It can be interpreted from the plot below that the general search takes less time as compared to the condition based search or specific search.

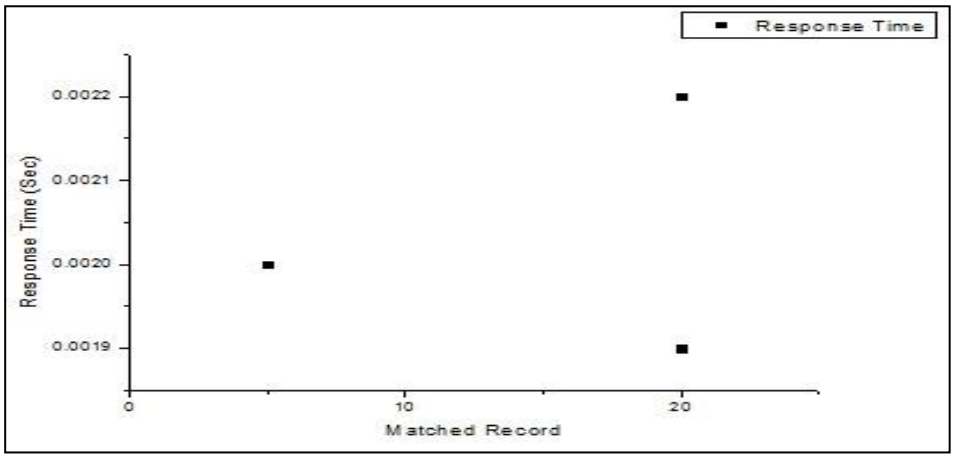


Figure 9.8: Specific Search Vs Normal Search

With general searched employed the response time taken to search 20 records is 0.0019sec and through book-id (condition based search) the response time is increased to 0.0022 sec.

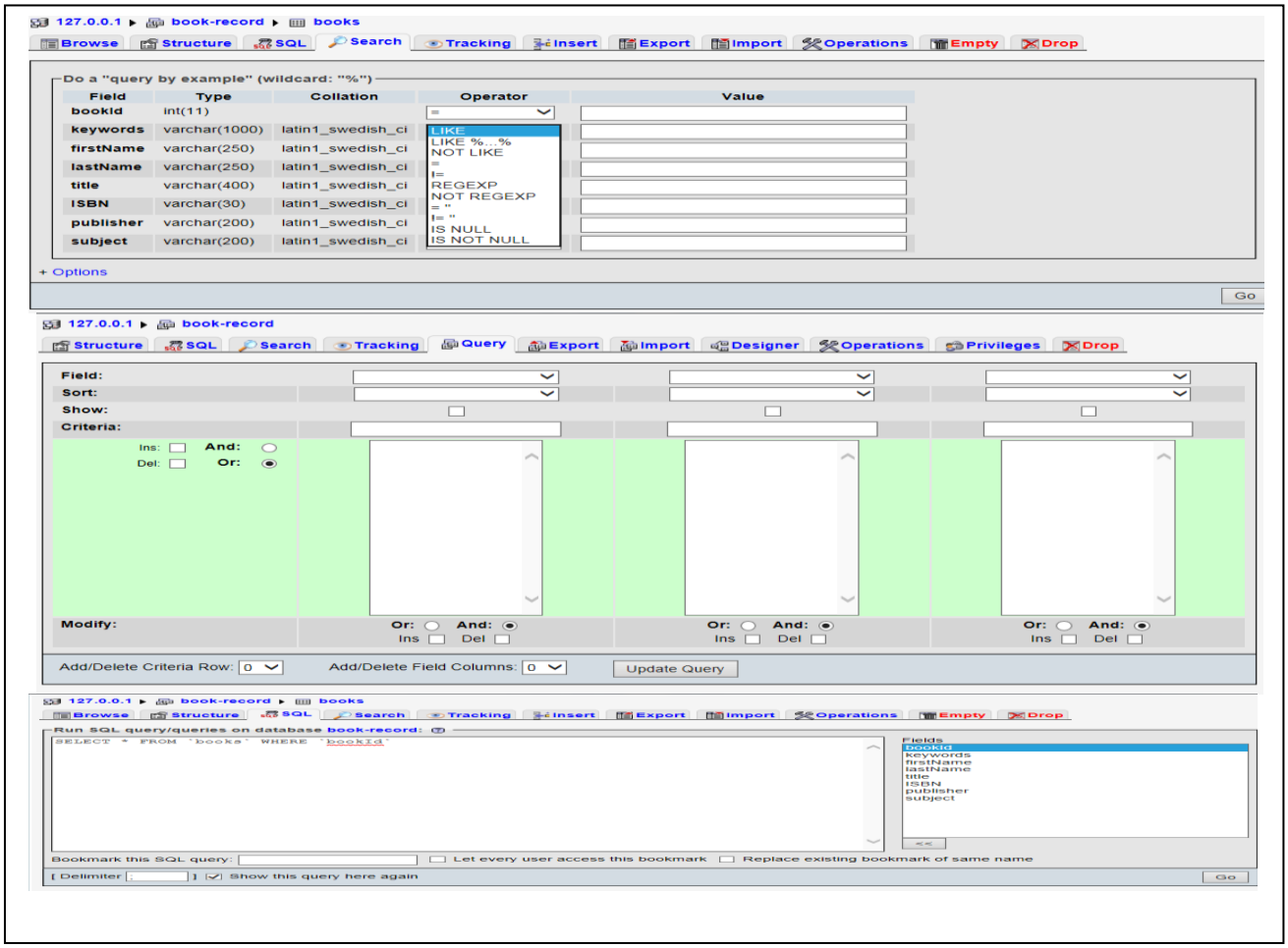


Figure 9.9: Condition apply on Process

Further using specific search to find 5 records takes 0.0020 sec which is still greater than the general search response time of 0.0019 sec. For the specific searching use like operator used shown in Figure 9.9.The proposed formula based calculation (completion time) is shown in Table 9.2.

Table 9.2: Expected Response Time

No. of Process	Matched Record	Expected Response Time	Searching Method
2	20	0.000095	All Information
2	5	0.0010	Specific Information
2	20	0.0011	Condition based Information
5	20	0.000038	All Information
5	5	0.00040	Specific Information
5	20	0.00044	Condition based Information

Figure 9.10 shows the graph relating expected response time with the total matched records. It can be seen from graph below that the proposed method of parallel working of hidden web crawler enhances the speed-up or efficiency of the system. The average response time taken to fetch the five records using single process was 0.0020 sec while with application of two processes the average time reduced to 0.0010 sec and with the addition of more processes says five the average response time is considerably reduced to 0.00040 sec.

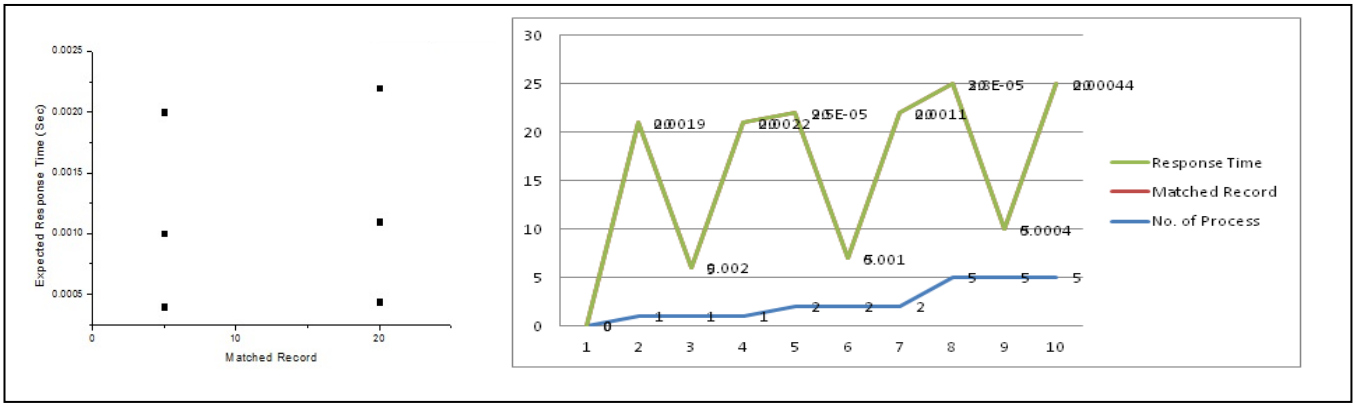


Figure 9.10: Expected Response time

The next section discusses the implementation of the second phase i.e. indexing of Hidden Web contents for efficient retrieval.

9.4 INDEXING OF HIDDEN WEB CONTENTS FOR EFFICIENT RETRIEVAL

The Schema Instance based Indexing refers to the piece of information in the form of column and rows (table), attributes taken as the origin of hidden web documents for a particular domain. The working of Schema Instance based on tree structure used to organize the domain and its attributes with values in cluster shown in Figure 9.11.

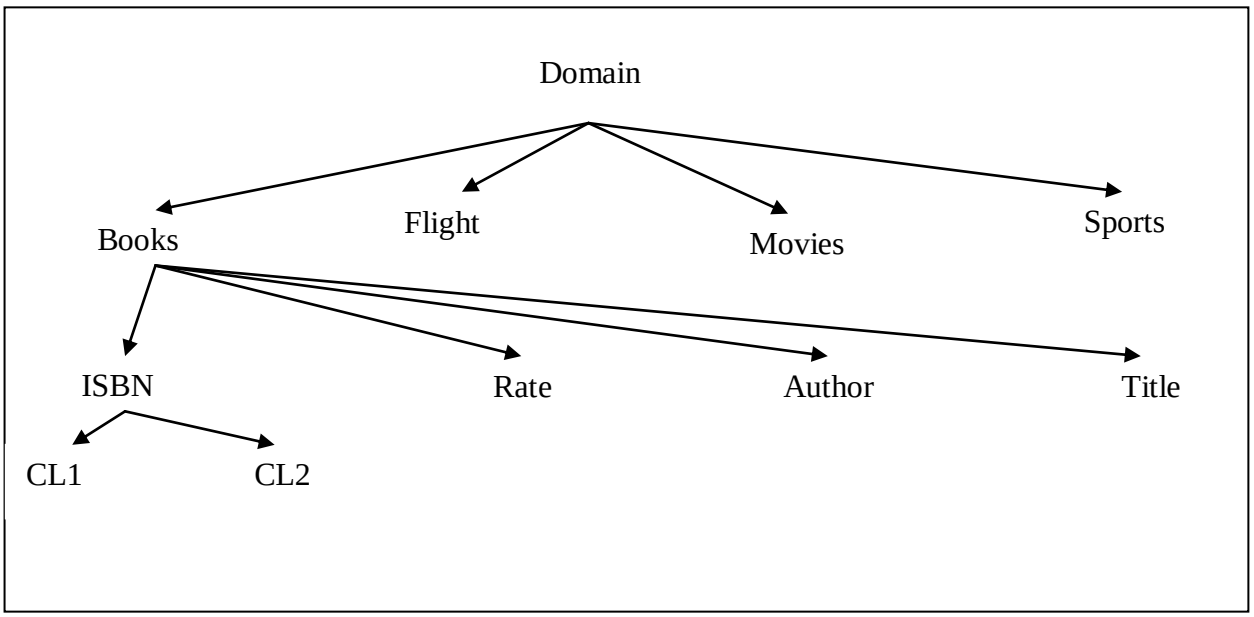


Figure 9.11: Organizing of Index

The benefits of the load balancer are given below:

- Avoid the synchronization of the data from the database.
- There is no Synchronization required.
- No inter-task communication delay.
- The concept of inter cluster communication improves the processing speed.
- Load Balancer distributes uniform records to each cluster

The Synchronization of the indexing is shown in Figure 9.12

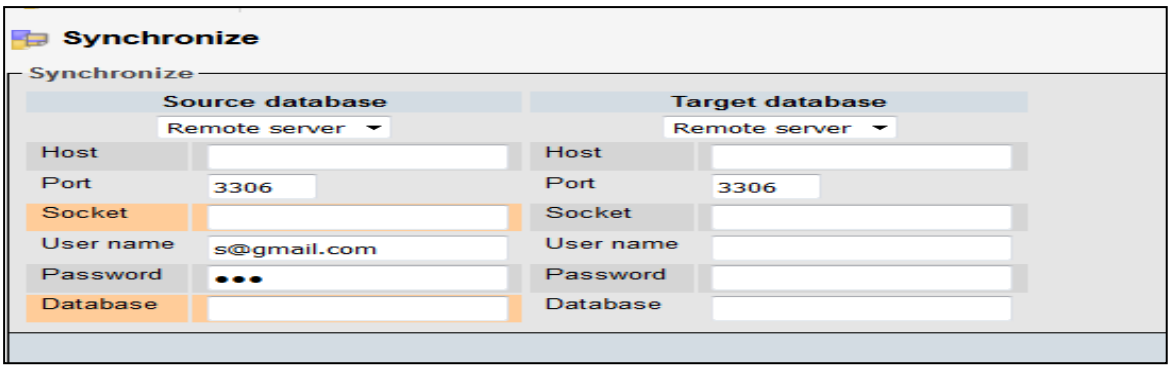


Figure 9.12: Synchronize port

The index in the Figure 9.13 in terms of book domain is created on the principle of BTREE. Initially Index is created on bookid with provision to add/delete attributes. The attribute bookid is of primary type and cardinality is 20 shown in Figure 9.13. The snap shot indexing shown in the Figure 9.13 below provides the primary information which is further segregated into the fields like type, collation, attributes etc. The action column shows whether the attribute has to be added or deleted in the table.

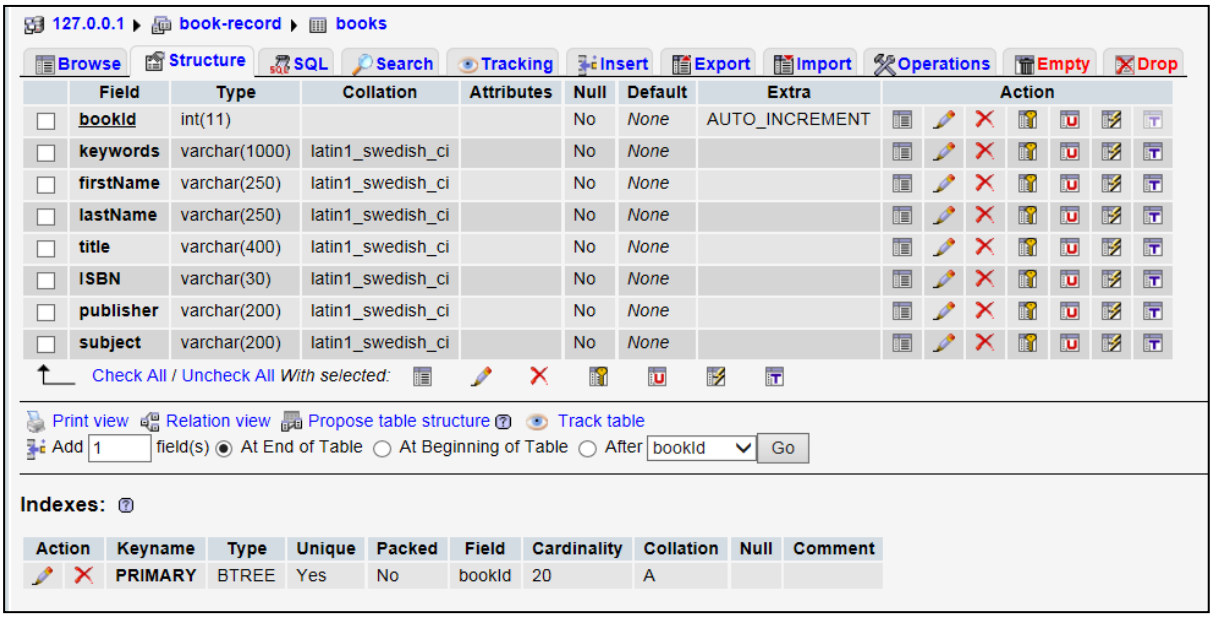


Figure 9.13: Snapshot Indexing

By clicking on the details tab, space usage parameters can be recorded and monitored.These details parameters of the index are shown in Figure 9.14. For example, in case of 20 records the space usages is 4000 Bytes and Index usage is 2048 Bytes . Other Row statistical details are also shown in the Figure 9.14 below.

- Details...

Space usage		
Type	Usage	
Data	4,000	B
Index	2,048	B
Total	6,048	B

Row Statistics	
Statements	Value
Format	dynamic
Collation	latin1_swedish_ci
Rows	20
Row length ø	200
Row size ø	302 B
Next Autoindex	21
Creation	Mar 16, 2015 at 08:21 PM
Last update	Mar 16, 2015 at 08:32 PM

Figure 9.14: Details of Index

When added more attributes to the indexing for obtaining mutiple ways of retrieval and finding the more relevant data from the page repository. Intially bookid is the only attribute for indexing there after two more attributes are added for indexing as shown in Figure 9.15.

Indexes:

Action		Keyname	Type	Unique	Packed	Field	Cardinality	Collation	Null	Comment
		PRIMARY	BTREE	Yes	No	bookId	20	A		
		keywords	BTREE	Yes	No	keywords	20	A		
		ISBN	BTREE	Yes	No	ISBN	20	A		

Figure 9.15: Add Index Attributes

Every time a new attribute is added in the index list, it takes few second to alter the relation. The Figure 9.16 shows the time query takes after each alteration to index list. When a single attribute title is added to the indexing, the query took 0.0862s as shown in Figure 9.16.

An index has been added on title (Query took 0.0862 sec)

```
ALTER TABLE `books` ADD UNIQUE (
  `title`
)
```

Figure 9.16: Add Index Attributes “TITLE”

Figure 9.17 shows the final output of the index after adding attribute title .The size of the index increased 2,048B to 12,288B .

Indexes:

Action		Keyname	Type	Unique	Packed	Field	Cardinality	Collation	Null	Comment
		PRIMARY	BTREE	Yes	No	bookId	20	A		
		keywords	BTREE	Yes	No	keywords	20	A		
		ISBN	BTREE	Yes	No	ISBN	20	A		
		title	BTREE	Yes	No	title	20	A		

Create an index on 1 columns Go

- Details...

Space usage		
Type	Usage	
Data	4,000	B
Index	12,288	B
Total	16,288	B

Row Statistics	
Statements	Value
Format	dynamic
Collation	latin1_swedish_ci
Rows	20
Row length ø	200
Row size ø	814 B
Next Autoindex	21
Creation	Mar 22, 2016 at 07:52 PM
Last update	Mar 22, 2016 at 07:52 PM
Last check	Mar 22, 2016 at 08:28 PM

Figure 9.17: Add title for indexing

Similarly if the any attribute title is dropped of the index list then according to the Figure 9.18 the query takes 0.1525sec.

Index username has been dropped (Query took 0.1525 sec)

```
ALTER TABLE `users` DROP INDEX `username`
```

Figure 9.18: Dropped Index Attributes

Finally it can be deduced from the above results that if the indexing is done on the basis of the keywords then the users get the opportunity of searching the relevant information in terms of versatile search terms. In case of book domain the user can search for a certain title by entering any of the fields like ISBN number, Author name, Title of the book etc. This also helps in removing the unwanted information being provided to the user in some cases. For instance the book author is Tanenbaum, who writes the books in the fields of computers and also literature. If the user only searches the books in the name of author he/she will be provided with both the contents in literature as well as computers. However the user only wishes to find the book authored by Tanenbaum in computers field only. In this case the multiple keywords concept will compare the author and the subject details simultaneously and would be able to filter out the relevant information for the user thereby reducing the time of search. Table 9.3 shows the values of index attributes, size of data in book domain and the variation of index size.

Table 9.3: Index Attribute

Index Attribute	Data	Index
1	4000	2048
2	4000	9216
3	4000	11264
4	4000	12888

Figure 9.19 shows the plot between index size and the Attributes. There is the steep rise in index size when the attribute is initially increased from 1 to 2 and thereafter the slope has significantly degraded as there is further rise in the index attributes.

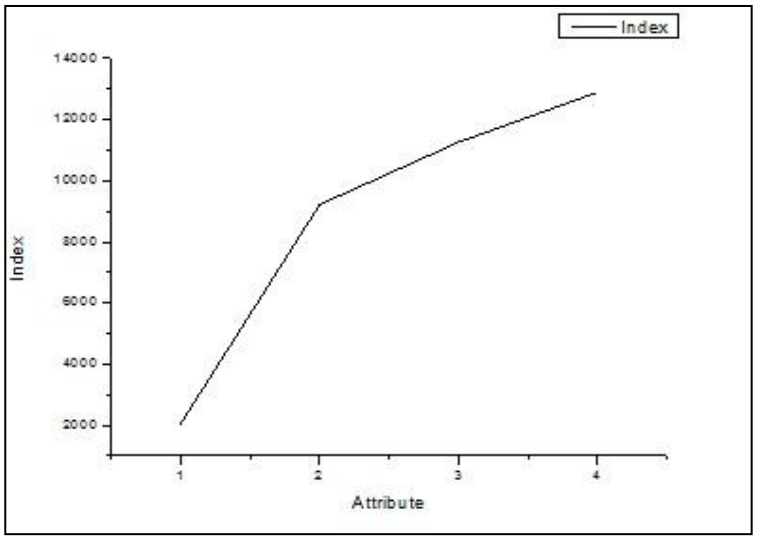


Figure 9.19: Index Attributes Graph

The next section discusses the implementation of the third phase i.e. domain specific query processing

9.5 DOMAIN SPECIFIC QUERY PROCESSING

When users fire the query for retrieving the information about the book domain, the query is generated for the search engine by the query processor. In query processing first step is linguistic modules, syntax checking and query modification. It then matches search term in the query syntax with views, relation and columns listed in the relation shown in Figure 9.6.It is obvious from the Figure 9.20 below that

Tokenization is the first step of linguistic module. In implementation tokenizer support is enabled and thus it removes the punctuations like comma, full stops etc from the query. WDDX is used as the term matcher in whom the documents are matched by applying the method of Boolean operations. Also it performs the task of session serializer.

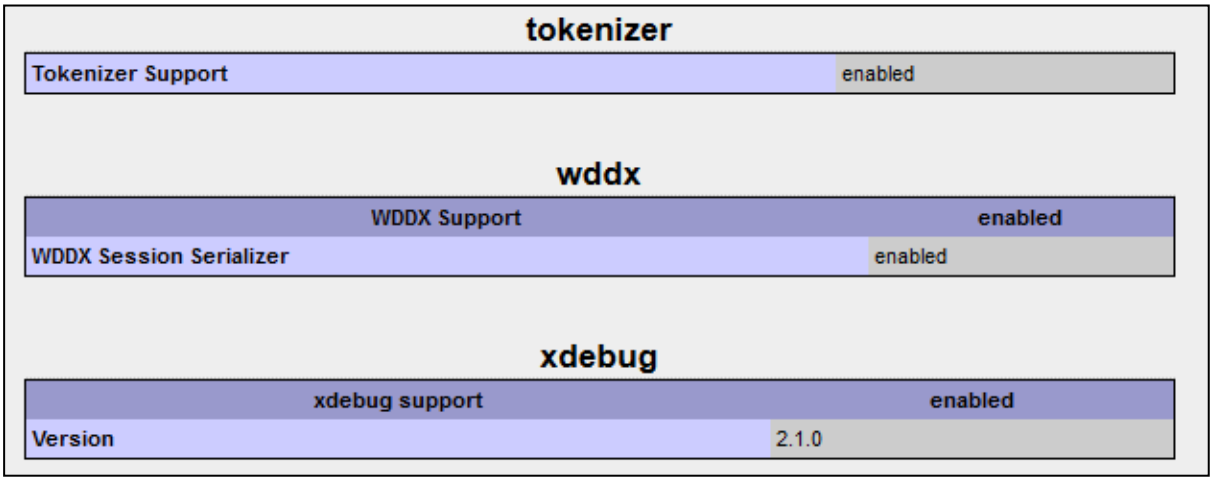


Figure 9.20: Tokenizer supports enable

In Figure 9.21 the search form of book domain is shown. Here, the user can find the books with different approaches like author name, book title, ISBN and through publisher name.

Figure 9.21: Search form Book Domain Database

Once the users click on the attributes like ISBN, Title, Publisher and Author name, all the available values belong to the particular attributes appear on the screen. The benefit of this technique is that the user takes less time to feed the data in search form shown in Figure 9.22 and Figure 9.23.

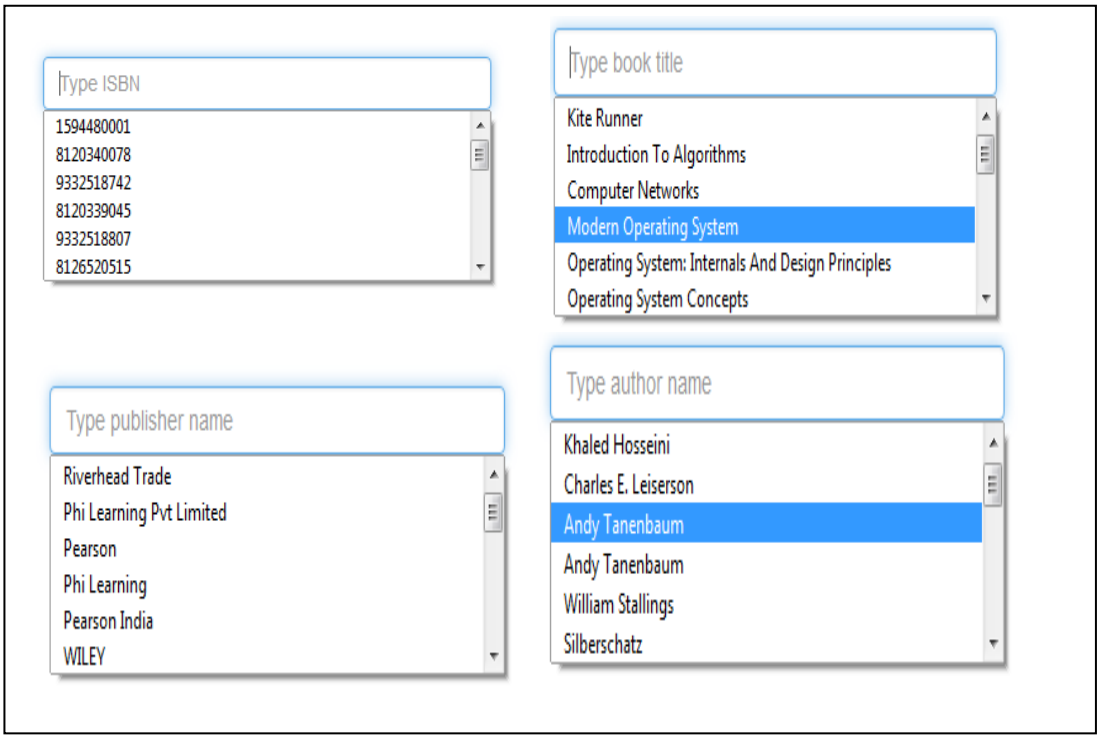


Figure 9.22: Automatic filling

Once a user choose the categories/attributes then identification of attributes contain by the database is carried out. It helps in carrying out more specific search related to particular author or title in case of book domain. This process enhances the pace of query processing.

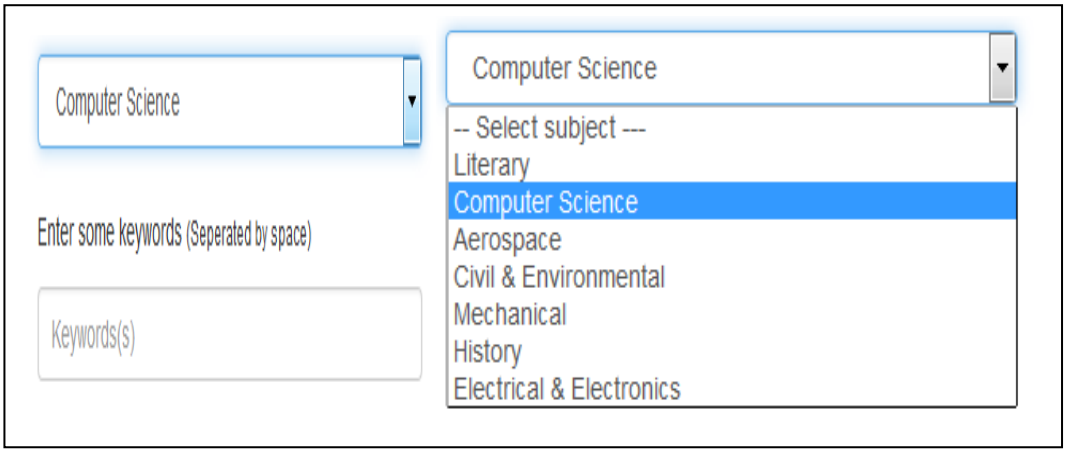


Figure 9.23: Select Subject for Search

According to the condition limit of 30 records specified in the proposed architecture, result page displays at a time 30 Records and the total cost to retrieve the more than 30 records set will be 30/30. It can be obtained in one communication round using the proposed model. For example as it shown in the Figure 9.24 below, if query is fired about the specific result says on subject computer science in book domain. The

proposed model fetches seven records as per query and the final value of records crawled is given as 7/20 records. In Figure 9.6, three different results are shown in it, first case the system will perform general searching and obtain the results much quicker as compared to the second and third case where the system retrieves 5 results out of 20 records and 20 results out of 20 records but takes longer time due to the system being involved more in performing matching tasks and generate the relevant output.

Result page

Author's First Name	Author's Last Name	Title	ISBN	Publisher	Subject	Keywords	Buy now
Charles	E. Leiserson	Introduction To Algorithms	8120340078	Phi Learning Pvt Limited	Computer Science	algorithms sorting quicksort augmenting	Buy now
Andy	Tanenbaum	Computer Networks	9332518742	Pearson	Computer Science	physical data link access control	Buy now
Andy	Tanenbaum	Modern Operating System	8120339045	Phi Learning	Computer Science	processes threads memory	Buy now
William	Stallings	Operating System: Internals And Design Principles	9332518807	Pearson India	Computer Science	memory management virtual exclusion	Buy now
Silberschatz		Operating System Concepts	8126520515	WILEY	Computer Science	deadlock file system	Buy now
Andrew	Appel	Modern Compiler Implementation In C/java	817596071X	CAMBRIDGE UNIVERSITY PRESS	Computer Science	scheduling optimization compiler design	Buy now
Hopcroft	Motwani And Ullman	Introduction To Automata Theory, Languages, And Computation	8131720470	Pearson India	Computer Science	computational complications	Buy now

Figure 9.24: Result Page Computer Science

The proposed query processing prototype helps to enhance the efficiency of query search mechanism by removing the unwanted data from the search interface thereby reducing the time of search. Table 9.4 enlists the parameters like search record, fetched record, precision, recall, accuracy and F-measure of search records of book domain.

Table 9.4: Efficiency Accuracy Table

S.No	**Search record**	**Fetched record**	**Available record in Database**	**Precision**
1	Computer Science	7	7	100%
2	History	3	3	100%

The next section discusses the implementation of the fourth phase i.e. ranking mechanism for domain specific hidden web.

9.6 RANKING MECHANISM FOR DOMAIN SPECIFIC HIDDEN WEB

The query generated by the user must be sent through the network to the destination for processing and then the result is sent back according to the ranking of the record. The database of book domain is shown in Figure 9.4. According to the proposed model, much of the search time is reduced by applying a very efficient mechanism of record ranking. In this mechanism the difference between the ranks of various records is computed and then the records are sorted in the manner in which the most popular record decided according to query log or the web log/buy log/search log is arranged first and so on. Buy log and search log are used in the implementation of the prototype. For example a user fires a query for searching the category of history in book domain which is shown Figure 9.25.

Figure 9.25: Select Subject for Searching

After the processing of the various steps like crawling, indexing, query processing, ranking process plays a very important role in reducing the search time. Figure 9.26 represents the output of the ranking technique in the tabular form, depicting the most popular history book at the top of table followed by the prioritised records.

Result page

Author's First Name	Author's Last Name	Title	ISBN	Publisher	Subject	Keywords	Buy now
Levi	Tillemann	The Great Race: The Global Quest For The Car Of The Future	1476773491	Simon & Schuster	History	agined. They will drive themselves, won't consume oil, and will come in radical shapes and sizes. But the path to that future is fraught The top contenders	Buy now
Mike	Mueller	The Complete Book Of Corvette - Revised & Updated: Every Model Since 1953 (Complete Book Series)	0760345740	Motorbooks	History	flagship sports car has become a timeless part of American culture and a household name across the	Buy now
A. J.	Baime	Go Like Hell: Ford, Ferrari, And Their Battle For Speed And Glory At Le Mans	0547336055	Mariner Books	History	omobile transportation to the masses was falling behind. Baby boomers were taking to the roads in droves, looking for speed not safety style not comfort	Buy now

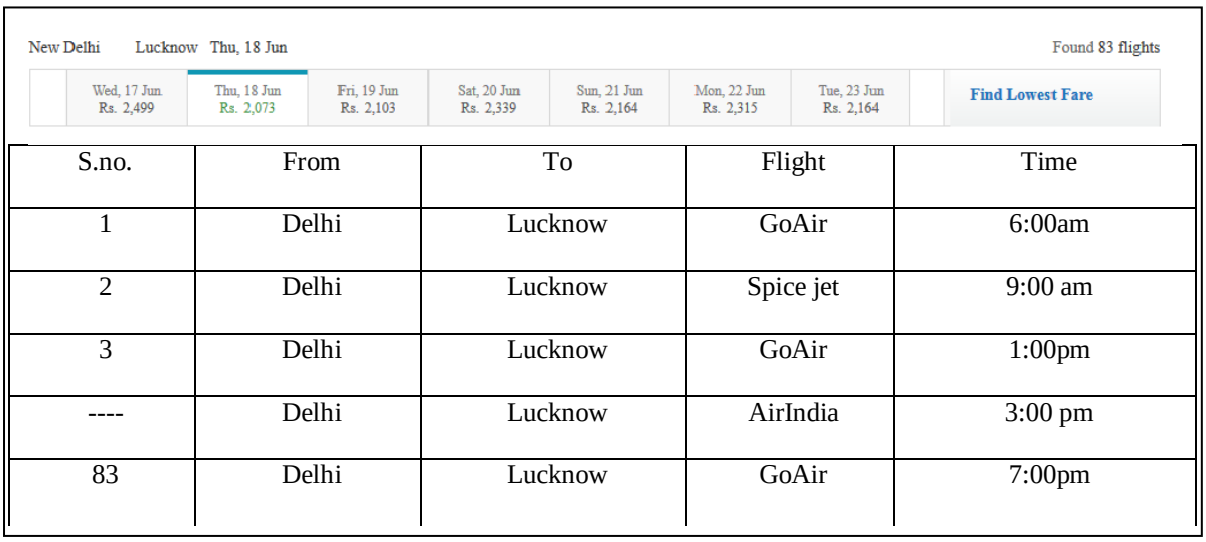
Figure 9.26: Result Page of Subject "History"

Table 9.5 shows the computed values of the results of score, ranking distance and tuple wise records in the book domain. This shows that in the proposed model the records are categorised on the basis of ranks distance as well as total score of the records resulting in assigning the top priority to the records with most number of clicks.

Table 9.5: Rank Calculation

Term	Search	Score	RD	RT
Subject	Computer Science	0.456	0.85	1.30
Publisher	PHI	0.824	0.428	1.25
Title	Operating System	0.824	0.29	1.114
Author	Andy Tanenbaum	1.0	0.14	1.14
All	Delhi Lucknow	0	0.99	0.99
GoAir	Delhi Lucknow	1.44	.0.99	2.43

Similarly the same model is applied to flight domain. Figure 9.27 shows the list of the records as per the query generated by the user for searching of particular record flight from Delhi to Lucknow. A total 83 records are listed in the table below.



S.no.	From	To	Flight	Time
1	Delhi	Lucknow	GoAir	6:00am
2	Delhi	Lucknow	Spice jet	9:00 am
3	Delhi	Lucknow	GoAir	1:00pm
----	Delhi	Lucknow	AirIndia	3:00 pm
83	Delhi	Lucknow	GoAir	7:00pm

Figure 9.27: Result Page Flight Domain

The Table 9.6 below shows the records listed on the basis of number of hits, as per the proposed model of record ranking.

Table 9.6: Record Rank Result

S.No	Possible Word Search	Result	Ranks
1	Computer Science	Domain Book	1st
2	PHI		2nd
3	Andy Tanenbaum		3rd
4	Operating System		4th
5	Go Air	Flight Domain	1st
6	All		2nd

The merits of the ranking technique can be summed up as proper sorting of the searched records as per the priority and thereby leading to the significant reduction in the search time and also brings considerable improvement in the accuracy of the system.

The next section discusses the implementation of the fourth phase i.e. predicting the next query for domain specific hidden web

9.7 PREDICTING THE NEXT QUERY FOR DOMAIN SPECIFIC HIDDEN WEB

Once the form is filled by the user, for instance the user wants to search books from books domain and flight to any destination. The Weblog maintains the relational database of the queries accessed by the user. Here client IP address plays very important role for the next query prediction. For example user searches for the flight ticket for a certain destination and books on the computer science category. For this query the user then visits a particular site as shown in the Figure 9.28.

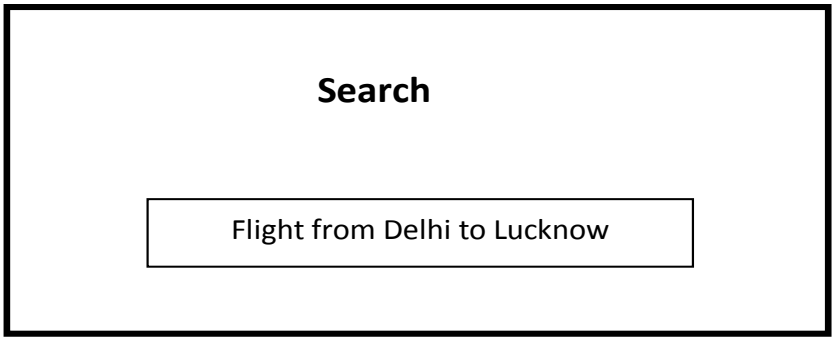


Figure 9.28: Search Form

First the data entered is cleaned, second the cleaned data is sent for the transaction layer. Data cleaning is a process to clean the unwanted symbol, punctuation, removal of the common and most frequent key words. For example, user fires a query to search any flight from Delhi to Lucknow before processing of the query the systems omits the unwanted values and symbols. Similarly in case of book domain unwanted value and symbol are omitted also. The users access the web pages according to the user preferences. The required information is stored in the form of columns which are important and are maintained in the weblog. In proposed prototype it is mandatory to specify IP address, Email Id, Uid to maintain the history of user access shown in Figure 9.29. Also emailid is required for the authentication of the user.

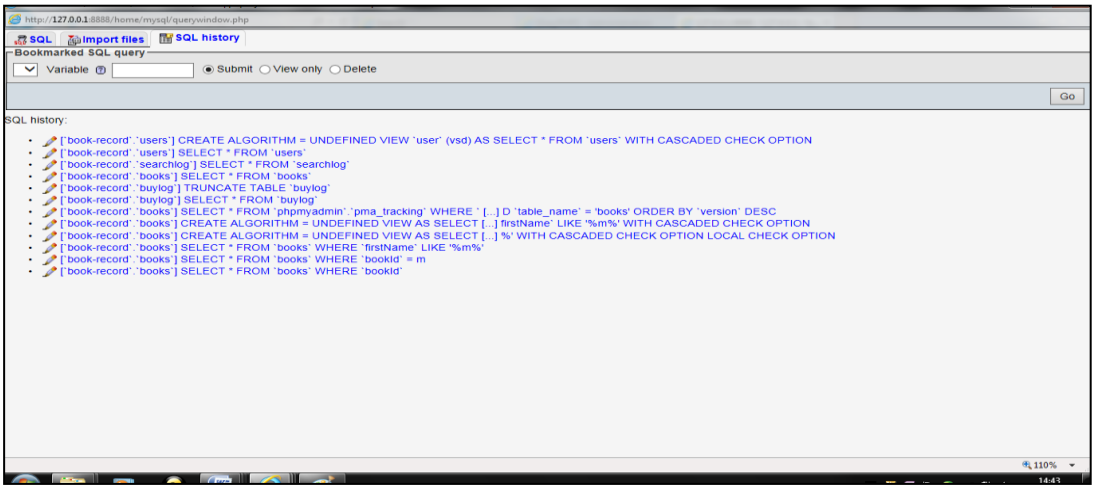


Figure 9.29: User History

Figure 9.30 shows the tables created by the proposed prototype for the implementation of collecting user history information. Also it specifies the time taken by the system to fetch a particular record as per the user access and the details of the user personal information.

uid	username	email	password
2	Test name	test@abc.com	abc
4	abcd	abcd@gmail.com	abcd
5	Sudhakar	sudhakarayush@gmail.com	ayush
6	Sudhakar Ranjan	sudhakarranjan@gmail.com	Ayush
7	sudhakar	sudhakar@gmail.com	aba
8	su	s@gmail.com	abc
9	s1@gmail.com	s1@gmail.com	abc

Figure 9.30: User Database

Transaction is identifying as the time spent by the particular user to access any site with IP address. Transaction process is confirmed after the value stored in buy log and search log shown in Figure 9.31.

session

Session Support	enabled
Registered save handlers	files user sqlite
Registered serializer handlers	php php_binary wddx

Figure 9.31: Session Support Enabled

Screen shot Figure 9.32 shows the implanted model output. This shows that if the administrator login in the system it can view any of the log files like search file and the buy log files consisting of the user search and purchases data.

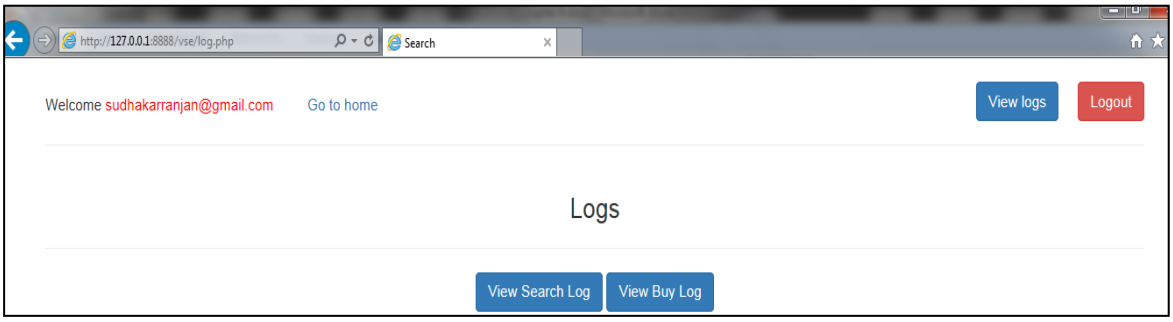


Figure 9.32: View Log Files

Query log/ search log contains the record of queries fired by different users at different time. It stores the database on the basis of the keywords of various domains and helps in the searching or finding the query.

Screen shot Figure 9.33 shows the search log history output of the prototype. This assigns a unique ID to the user which login in particular session every time. It also represents the book id s searched by the user at any time.

searchlogid	email	bookid
1	abcd@gmail.com	2
2	abcd@gmail.com	3
3	abcd@gmail.com	4
4	abcd@gmail.com	5
5	abcd@gmail.com	6
6	abcd@gmail.com	7
7	abcd@gmail.com	8
8	abcd@gmail.com	2
9	abcd@gmail.com	3
10	abcd@gmail.com	4
11	abcd@gmail.com	5
12	abcd@gmail.com	6

Figure 9.33: Search Log File

Screeen shot Figure 9.34 displays the following results computed from the proposed prototype model. Firstly the XML code for the buylog history is given.


```
<?xml version="1.0" encoding="UTF-8"?>
<!-- - phpMyAdmin XML Dump - version 3.3.9.2 - http://www.phpmyadmin.net - - Host: 127.0.0.1 - Generation Time: Mar 22, 2016 at 02:43 PM - Server version: 5.5.10 - PHP Version: 5.3.6
-->
<pma_xml_export xmlns:pma="http://www.phpmyadmin.net/some_doc_url/" version="1.0">
        <!-- - Structure schemas -->
  - <pma:structure_schemas>
      - <pma:database charset="latin1" collation="latin1_swedish_ci" name="book-record">
            <pma:table name="buylog"> CREATE TABLE `buylog` ( `buylogid` int(11) unsigned NOT NULL AUTO_INCREMENT, `email` varchar(100) NOT NULL, `bookid` int
              (11) NOT NULL, PRIMARY KEY (`buylogid`), UNIQUE KEY `buylogid` (`buylogid`) ) ENGINE=InnoDB AUTO_INCREMENT=7 DEFAULT CHARSET=latin1;
              </pma:table>
        </pma:database>
    </pma:structure_schemas>
        <!-- - Database: 'book-record' -->
  - <database name="book-record">
            <!-- Table buylog -->
      - <table name="buylog">
            <column name="buylogid">1</column>
            <column name="email">sudhakarranjan@gmail.com</column>
            <column name="bookid">3</column>
        </table>
      - <table name="buylog">
            <column name="buylogid">2</column>
            <column name="email">sudhakarranjan@gmail.com</column>
            <column name="bookid">2</column>
        </table>
```


Figure 9.34: XML Buy Log File

In second result the buylog history in tabular form using SQL code shown in Figure 9.35.The buylog history table assigns the unique id to the user according to the session and maintains the book id records of the same/different user for different sessions. Also the emailid is stored in order to compute the popularity of the items listed in the buy log table which useful for perfoming the task of next query prediction.

buylogid	email	bookid
1	sudhakarranjan@gmail.com	3
2	sudhakarranjan@gmail.com	2
3	sudhakarranjan@gmail.com	3
4	sudhakarranjan@gmail.com	2
5	s1@gmail.com	13
6	s@gmail.com	9

Figure 9.35: Buy Log Database

The snap shot of user previous buy log items is shown in Figure 9.36 and Figure 9.37. This provides the user freedom to view the past purchases by logging into the Figure 9.36.

Buy Logs

User email	Book id
sudhakarranjan@gmail.com bought book with id	3
sudhakarranjan@gmail.com bought book with id	2
sudhakarranjan@gmail.com bought book with id	3
sudhakarranjan@gmail.com bought book with id	2
s1@gmail.com bought book with id	13
s@gmail.com bought book with id	9

Figure 9.36: Buy Log File

Lastly the transaction image is shown in Figure 9.37, which enables the user to make transaction.

Welcome sudhakarranjan@gmail.com Go to home View logs Logout

Verify cart and do payment

Book name : The Complete Book Of Corvette - Revised & Updated: Every Model Since 1953 (Complete Book Series) with ISBN 0760345740

Proceed to payment $

Figure 9.37: Transaction

The screen shot in the Figure 9.38 shows response time and other features of Query log/ search log/User. The Figure9.38 shows the response time taken by the system to represent the user records in different log file like search log, buy log and book records.

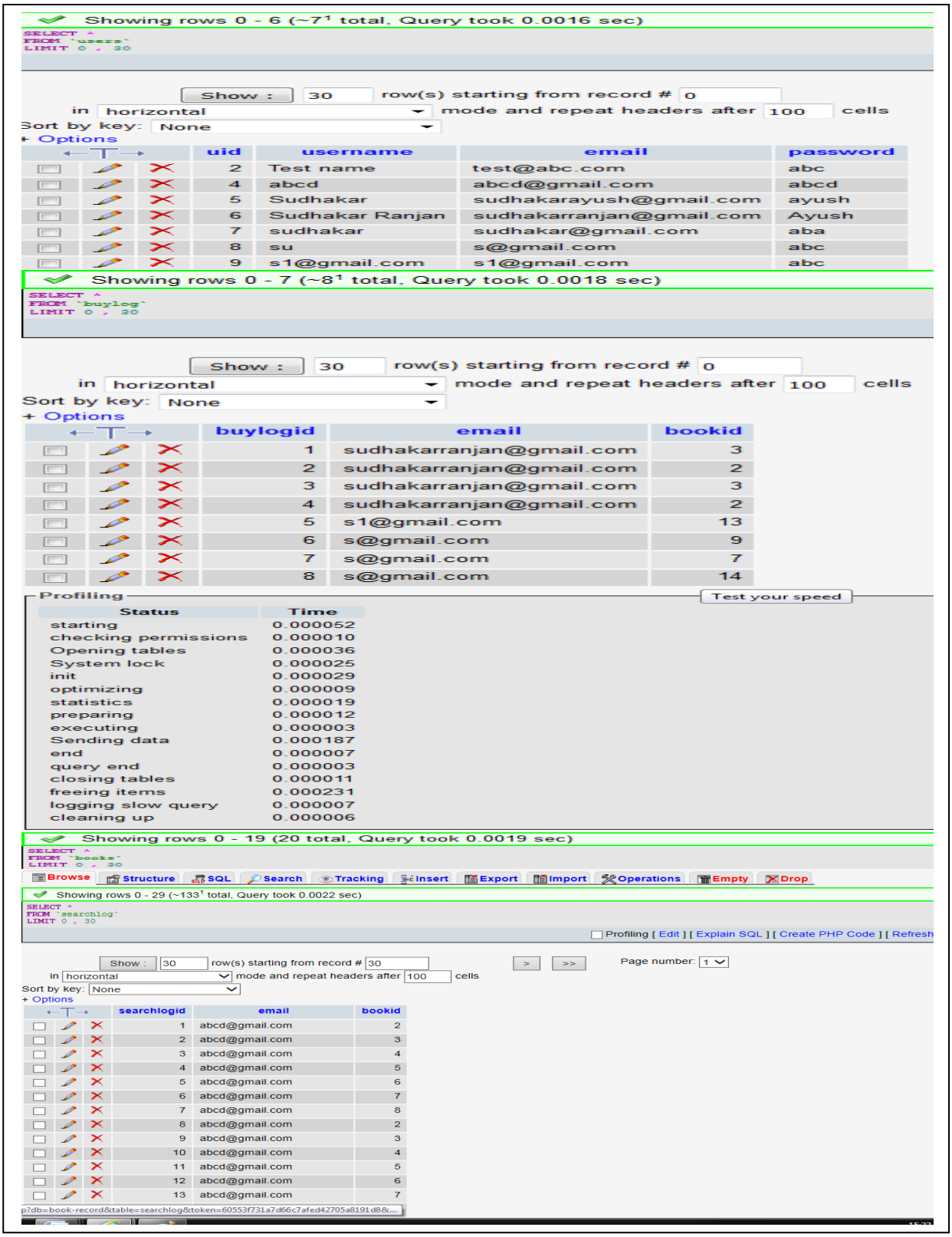


Figure 9.38 Response time User, Buylog, Search log

The table [9.7] shows the reponse time of records represented by user, buy log, books and search log. The reponse time increases linearly as the number of matched records and found records are increasing. Using the multiple instance technique in the proposed model will further reduce the response time.

Table 9.7 Found Record & Response Time

S.No.	Particulars	Found Record	Response Time
1	User	7	.0016
2	Buylog	8	.0018
3	Books	20	.0019
4	Search Log	233	.0022

Figure 9.39 shows the plot of the Table 9.7, This shows the relationship between the different searched particular items with matched records and the response time taken by the system.

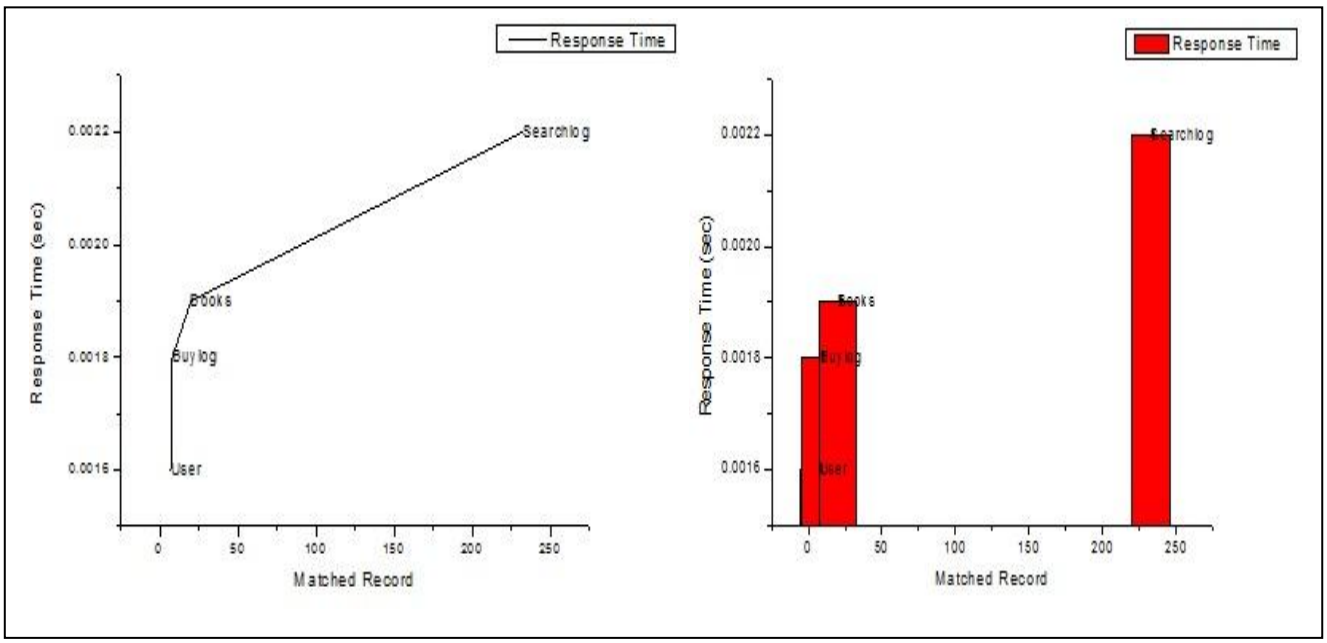


Figure 9.39 Matched Record vs Response time

Table 9.8 compares the proposed Least Cost Vertical search engine based on DSHWC over certain parameters with some of the existing crawlers for the Hidden Web.

Table 9.8: Comparison of Proposed Architecture with the Existing Crawlers

S.No	Deep WebCrawler/Hidden Web Crawler	Load Sharing	Maximum Throughput	Record Ranking	Load Balancing	Technique
1	A Framework of Deep Web Crawler	No	No	No	No	I. 1.Form Analysis II. 2.Value assignment and submission III. 3.Response analysis IV. 4.Response Navigation
2	AKSHR: A Novel Framework for a Domain-specific Hidden Web Crawler	No	No	No	No	I. 1.Search Interface Crawling II. 2.Domain-specific Interface Mapping III. 3.Automatic Form Filling IV. 4. Response Page Analysis
3	Finding the WDB's Query Interface in Deep Web Automatically	No	No	No	No	I. Extract Features of Query Forms II. 2. Extracting Query Interface Features of Domain Query Interfaces III. 3. Identifying and Classifying Query Interfaces
4	Hidden Web Database Exploration	No	No	No	No	I. 1.Hidden Web Pages II. 2.Parse Web Pages III. 3.Generate the Knowledge Base for a specific domain IV. 4.Analyse the Response Pages V. 5.The Relevance of a hidden database
5	A Crawler for Local Search	No	No	No	No	I. 1.Determine Relevancy II. 2. Crawling Behaviour
6	Domain-specific Web Service Discovery with Service Class Descriptions	No	No	No	No	I. 1.Search Form Filter II. 2.Form Interface Analyzer III. Module Selection IV. Query Generator V. Query Selection &

						VI.	Probing Response Matching
7	A Vertical Search Engine – Based On Domain Classifier	No	No	No	No	I. II. III. IV. V. VI. VII.	Web Crawler HTML Parser Filter Domain Classifier Page Ranker URL DB Search
8	A Distributed Approach To Crawl Domain Specific Hidden Web	No	No	No	No	I. II. III. IV.	Analyser Parse Composer Result Analyzer
9	Crawling the Hidden Web	No	No	No	No	I. II. III. IV.	Classify Dynamic Web Content Modeling Forms and Form Submissions HiWE: Hidden Web Exposer Design Issues and Techniques
10	Exploiting Ontology for Retrieving Data Behind Searchable Web Forms	No	No	No	No	I. II. •	Response Analysis Form Processing Form Analysis Page Classification Matching with ontology Form filling & automatic query generation
11	How Search Engines Work and a Web Crawler Application, Dept of Computer science University of lllinious at Springfield, IL62703	No	No	No	No	I. II. III. IV.	Working of search engine Working of web crawler Meta search engine Indexing
12	Proposed Least Cost Vertical Search	Yes	Yes	Yes	Yes	I. II.	Search Interface Crawling Domain-specific Interface

	Engine based on DSHWC					Mapping III. Automatic Form Filling IV. Response Page Analysis Ref(AKSHR: A Novel Framework for a Domain-specific Hidden Web Crawler)

The productivity of the proposed least cost vertical search engine based on DSHWC, is specified on the basis of calculation well known parameters like *Precision, Accuracy, Recall* and *F-measure*. The computed results of the above mentioned parameters were found to be higher compared to the existing Hidden Web Crawlers. The proposed architecture also optimizes the network bandwidth using load balancers. Also the response time is reduced significantly through functions like indexing, ranking, query processing and next query prediction.

Chapter X

CONCLUSION & FUTURE SCOPE

10.1 CONCLUSION

In this dissertation, an effective method to collect the data with least response time from the Hidden Web has been developed. The main challenges involved in the task have been addressed and resolved.

During this work, many existing system of Hidden Web crawling with various components like indexing, query processing, ranking and next query predictions were studied with a view to understand the existing Hidden Web crawling mechanism and some unresolved issues found thereof were addressed and resolved as follows.

i. **Hidden Web Crawling:** Prior work in this field has not focused on all the aspects related to the Hidden Web like load sharing, multiple instance Hidden Web crawling. Therefore, it has being analyzed that there is a need to design and develop a reliable, scalable and fault tolerance rugged system.

ii. **Indexing the Hidden Web Content:** The accessible structures has not alert on the indexing of the web pages for Hidden Web crawling. Thus, there is a need to design efficient indexing mechanism for obtaining accurate results with minimum response time has been proposed and implemented. The Index constructor and Index search algorithm for the Hidden Web has been proposed. A load balancer approach in indexing technique has been developed. It helps to distribute the values of the attributes among different clusters attached with the system thereby obtaining accurate and high precision results.

iii. **Effective Vertical Web Access:** The huge size and heterogeneity of the Vertical Web makes complete coverage very difficult. Thus, in this work, Query processing and Record Ranking has been proposed and implemented. The proposed Query processing and Record Ranking leads to faster retrieval of the search information and thus adding to the further reduction of the response time.

iv. **Web Intelligence**: A mechanism automatically discovers the previous search, based upon web log and query log analysis has been proposed and implemented.

The Least Cost Vertical Search Engine based on DSHWC has been implemented using PHP and MYSQL. For calculating a least cost, various tests have been conducted on Domain Specific Hidden Web Crawler (DSHWC) and these test results indicate that the proposed LCVSE efficiently crawls the hidden web pages. The following components of proposed architecture like multiple instances of hidden web crawler, load sharing architecture, indexing, query processing, record ranking and intelligent web (next query prediction) plays an important role in reducing the cost in term of response time/crawling time and maximum utilization of network bandwidth.

10.2 FUTURE SCOPE

In this dissertation, the problems related to Hidden Web crawling have been determined broadly. Some of the possible future research in this area can be as follows:

i. **Intelligent Question Answering System:** The processing from the web document and image machines is able to categorize and recognize from the various fields like text, place, face, and people etc. There are large volumes of text, image and video data available on the web. To learn from the domain specific or vertical web it requires the intelligent memory and predictive intelligence for smart questioning and answering system.

ii. **Social Intelligence and Business Intelligence:** The size of the web contents is increasing day by day, through the application of web intelligence techniques. Also artificial intelligence techniques can be applied for the design of a social and business intelligent system.

REFERENCES

[1] KJ (Ken) Salchow, Load Balancing 101: Nuts and Bolts: White Paper 2015, F5 Networks, Inc.

[2] How AWS Pricing Works July 2014, http://aws.amazon.com/whitepapers/

[3] Steve Pederson ,Understanding the Deep Web in 10 Minutes : White Paper 2013 www.brightplanet.com

[4] MICHAEL K. BERGMAN The Deep Web: Surfacing Hidden Value White Paper 2001 www.brightplanet.com

[5] Iffat, Rabia and Sami, Lalitha K., "Understanding the Deep Web" (2010). Library Philosophy and Practice (e-journal). Paper 364. http://digitalcommons.unl.edu/libphilprac/364

[6] 90di.com – A travel Search White paper, 2009

[7] History of Search Engines: From 1945 to Google Today› General Discussion, Sep 29, 2011, www.scionxbforum.com

[8] Emanuele Tarantino, Vertical Search Engines Foreclosure, October 2011

[9] Sri khetwat Saritha,Kishan Dharvath, Domain and Keyword Specific Data Extraction From Invisible Web Databases, Eighth International Conference on Information Technology: New Generations: IEEE 2011.

[10] Bing Zhou,Xiao, Zhiqing Lin, Chuang Zhang: A Distributed Vertical Crawler Using Crawling-Period Based Strategy: IEEE 2010

[11] Min Lil, Jun Zhaol, Tinglei Huang, Research and Design of the Crawler System in a Vertical Search Engine, IEEE 2010

[12] Lovesh Kumar Desai, A Distributed Approach To Crawl Domain Specific Hidden Web, Georgia State University 2007

[13] Monica Peshave : How Search Engines Work and a Web Crawler Application",Dept of Computer science University of lllinious at Springfield,IL62703

[14] Xiang Peisu, Tian Ke, Huang Qinzhen: A Framework of Deep Web Crawler,Proceedings of the 27th Chinese Control Conference July 16-18, 2008, Kunming,Yunnan, China.

[15] Komal kumar Bhatia, A.K.Shrma, Rosy Madaan: AKSHR: A Novel Framework for a Domain-specfic Hidden web crawler, Proceeding of the 1st

international Conference on Parallel, Distributed and Grid Computing (PDGC-2010)

[16] Peiguang Lin, Ruzhi Xu, Zhimin Hong, Yan Zhang, Finding the WDB's Query Interface in Deep Web Automatically, IEEE International Conference on Internet Computing in Science and Engineering, 2008.

[17] Pedro Huitema,Perry Fizzano: A Crawler for Local Search, 2010 Fourth nternational Conference on Digital Society, IEEE 2010

[18] Rajashree Shettar, Rahul Bhuptani: A vertical Search Engine –Based On Domain Classifier, International Journal of Computer Science and Security, Volume (2) : Issue (4), 2008

[19] S. Raghavan, H. Garcia-Molina. Crawling the Hidden Web, in: Proc. of the 27th Int. Conf. on Very Large DataBases (VLDB 2001), September 2001.

[20] A.IEL-desoky, A.O.AbdEl-Gwad, M.E okasha: Exploiting Ontology for Retrieving Data BehindSearchable Web Forms (IEEE 2009)

[21] Zhiguo Gong, Jingbai Zhang, Qian Liu: Hidden Web Database Exploration.IEEE 2006.

[22] Jayant Madhavan, Loredana Afanasiev, Lyublena Antova, Alon Halevy Harnessing the Deep Web: Present and Future :,CIDR perspective 2009.

[23] Jianfeng Zheng, Zaiqing Nie: Architecture and Implementation of an Object-level Vertical Search, 2009 International Conference on New Trends in Information and Service IEEE 2009.

[24] Tiezheng Nie, Derong Shen , Ge Yu ,Yue Kou, Subject-Oriented Classification based on Scale Probing in the Deep Web Ninth IEEE International Conference on Web-Age Information Management 2008.

[25] Ahmad Ibrahim, Syed Ahsan Fahmi, Sajid Ibrahim Hashmi, and Ho-Jin Choi, Addressing Effective Hidden Web Search Using Iterative Deepening Search and Graph Theory IEEE 8th International Conference on Computer and Information Technology Workshops 2008.

[26] Ankur Narang, Karthik Swaminathan, Prashant Agrawal, Performance Optimizations for Distributed Real-time Text Indexing IBM India Research Laboratory, New Delhi, India,IEEE 2009,

[27] Ankur Narang, Vikas Agarwal, Monu Kedia and Vijay K Garg, Highly Scalable Algorithm For Distributed Real-Time Text Indexing:IBM India Research Laboratory, New Delhi, INDIA,IEEE 2009.

[28] Yong Zhang,Jian-lin li ,Research and Improvement of Search Engine based on Lucence: IEEE 2009.

[29] LiWen –Ze, Xu Tao, Research and Design of an efficient Chinese Indexing System: IEEE-2010

[30] Thomas Paul, The licence search engine : http:// Jakarta.apache.org/lucene

[31] Sharadh Ramaswamy, Kenneth rose, Towards Optimal Indexing for Relevance Feedback in Large Image database IEEE TRANSACTION ON IMAGE PROCESSING VOL 18 N0.12 2009

[32] Zhisheng Li, Ken C.K.Lee, Baihua Zheng. IR-Tree: An Efficient Index for Geographic Document Search, IEEE 2011

[33] Bo Geng, Member, IEEE, Linjun Yang, member, IEEE, Chao Xu, Xian-Sheng Hua, Ranking Model Adaptation for Domain-Specific Search, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, IEEE 2010.

[34] Bo Geng, Member, IEEE, Linjun Yang, Member, IEEE, Chao Xu, Xian-Sheng Hua, Member, Ranking Model Adaptation for Domain-Specific Search, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING 2012

[35] Kai Zheng, Zi Huang, Aoying Zhou Member, IEEE, and Xiaofang Zhou: Discovering the Most Influential Sites over Uncertain Data: A Rank Based Approach, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING: IEEE 2011.

[36] Jeffrey Jestes, Graham Cormode, Feifei Li, and Ke Yi : Semantics of Ranking Queries for Probabilistic Data, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING:IEEE2010

[37] Huang Decai, QI Huachun, Yuan, Zheng Yue-Feng. TC-PageRank Algorithm Based on Topic Correlation[C]. Proceedings of the 6th World Congress on Intelligent Control and Automation, June 21-23,2006, Dalian, China.2006:5943-5946.

[38] Yue kou,Derong Shen,Ge Yu,Tiezheng Nie.LG-ERM :An Entity –Level Ranking Mechanism for Deep Web query.Ninth International conference on Web –Age Information Management: IEEE 2008.

[39] Qi Hua-chun, Huang De-cai, Zheng Yuefeng,An Improved PageRank Algorithm with Time Feedbacking . Journal of Zhejiang University of Technology,2005

[40] WWW.iprcom.com/papers/pagerank/

[41] George Valkanas, Alexandros Ntoulas, Dimitrios Gunopulos, Rank-Aware Crawling of Hidden Web sites: Fourteenth international workshop Web DB2011

[42] Hesham Ali, Self ranking and evaluation Approach for Focused Crawler Based on Multi- Agent System, The international Journal of Information Technology,Vol 5,No,2 April 2008.

[43] Ling Zheng,YangBo,Ning Zhang,An Improved link Selection Algorithm for Vertical Search Engine,ICISE 2009.

[44] Ling Zheng,YangBo,Ning Zhang, Link – Sensitive Page Rank: An Improved Ranking Algorithm for Vertical Search Engines,IEEE 2009.

[45] Santoshi Oyama,Query Refinementfor Domain Specific Web Search, Grant Aid for Scientific Research,1999

[46] Michael Chau,Hsinchum Chen, Comparisonof three vertical search spiders,IEEE 2003.

[47] Lin-Tao,Li-PingHong –Fang Zhou, An Improved Topic Relevance Alogorithm For vertical Search Engine, IEEE 2008

[48] Yongmei LUO,Zhigang JIN, Ximan ZHAO: Design and Implementation of a Vertical Search System for Course Materials, IEEE 2010

[49] Suke Li, Zhong Chen, Liyong Tang, Zhao Wang: TSearch: A Self-learning Vertical Search Spider for Travel, IEEE 2008

[50] Lei Zhang, Yong Peng, Xiangwu Meng, and Jie Guo: Personalized Domain-specific Search Engine: IEEE 2008

[51] Xuan-Hieu Phan, Cam-Tu Nguyen, Dieu-Thu Le, Le-Minh Nguyen, A Hidden Topic-Based Framework toward Building Applications with Short Web Documents :2011

[52] Lei Yan, Ting wang,Yang Sang, A Research on Theme Correlation of Vertical Search Engine Based on Ontology, 2010 International Conference on Information, Networking and Automation (ICINA):IEEE 2010

[53] Bao-hua Qiang, Jian-qing Xi, Effective Schema Extraction of Query Interfaces on the Deep Web, Fifth IEEE International Conference on Fuzzy Systems and Knowledge Discovery, 2008.

[54] Ying Wang, Wanli Zuo, Tao Peng, Fengling He, Integration of Query Interfaces for Deep Web Databases, Fourth IEEE International Conference on Natural Computation 2008.

[55] Ping Wu, Ji-Rong Wen, Huan Liu, Wei-Ying Ma, "Query Selection Techniques for Efficient Crawling of Structured Web Sources

[56] Stefan Endrullis,Andreas thor, Rahm, Evaluation of Query Generators for Entity Search Engines,USETIM 2009

[57] Xin Zhong, Yuchen Fu, Quan Liu, Xinghong Lin, Zhiming Cui, A Holistic Approach on Deep Web Schema Matching, IEEE International Conference on Convergence Information Technology, 2007.

[58] Komal Kumar Bhatia, A.K. Sharma, "AKSHR: A Novel Framework of a Domain-specific Hidden Web Crawler", IEEE International Conference of Advanced Computing 2009.

[59] Komal Kumar Bhatia, A.K.Sharma, Neelam Duhan, "Page Ranking Algorithms: A Survey", IEEE International Conference of Advanced Computing 2009.

[60] Komal Kumar Bhatia, A.K.Sharma, "A Task-specific Hidden Web Crawler", CIC 2008, 17th International Conference on Computing,December 3 to5, 2008 Mexico City, Mexico.

[61] Komal Kumar Bhatia, A.K.Sharma, "A Framework for an Extensible Domain specific Hidden Web Crawler (DSHWC)", IEEE TKDE Journal.2008

[62] Komal Kumar Bhatia, A.K.Shrma: A Frame Work for Domain Specific Interface Mapper (DSIM), IJCSNS International Journal of Computer Science and Network Security, VOL.8 No.12, December 2008

[63] Komal Kumar Bhatia, A.K.Sharma, "Merging Query Interfaces in Domain-specific Hidden Web Databases" in an International Journal in 2008.

[64] K. Sharma, Komal Kumar Bhatia: "Automated Discovery of Task Oriented Search Interfaces through Augmented Hypertext Documents" Proc. First International Conference on Web Engineering & Application (ICWA2006).

[65] Luciano Barbosa and Juliana Freire. Searching for hidden-web databases. In Proceedings of WebDB'05, pages 1–6, 2005.

[66] Luciano Barbosa and J. Freire, Siphoning hidden-web data through keyword-based interfaces. In SBBD, 2004.

[67] Luciano Barbosa and Juliana Freire. An adaptive crawler for locating hidden-web entry points. In Proceedings of the 16th international conference on World Wide Web, pages 441–450, 2007.

[68] Luciano Barbosa and Juliana Freire. Combining classifiers to identify online databases. In Proceedings of WWW'07, pages 431–440, 2007

[69] Zhang, Z., He, B., C.-C. Chang, K.: Light-weight Domain-based Form Assistant: Querying Web Databases On the Fly. In Proceedings of the 31st Very Large Data Bases Conference, 2005.

[70] Sudhakar Ranjan, Komal Kumar Bhatia,"Query Interface Integrator for Domain Specific Hidden Web" International Journal of Computer Engineering & Applications Vol IV, Issue I/III, Oct.-Dec. 2013, Indexed by DBLP**,** arxiv.org.

[71] Sudhakar Ranjan, Komal Kumar Bhatia, Indexing for Vertical Search Engine: Cost Sensitive" International Journal of Emerging Technology & Advanced Engineering Volume 3, Issue 10, October 2013.

[72] Sudhakar Ranjan, Komal Kumar Bhatia, "Indexing for Domain Specific Hidden Web" International Journal of Computer Engineering & Applications Vol VII, Issue I, July 2014.

[73] Sudhakar Ranjan, Komal Kumar Bhatia, "Web Log for Domain Specific Hidden" International Journal of Computer Engineering & Applications Vol VII, Issue I, July 2014.

[74] Sudhakar Ranjan, Komal Kumar Bhatia, "Transaction in Hidden Web" International Journal of Computer Engineering & Applications, Vol VIII, Issue II, November 2014.

[75] Sudhakar Ranjan, Komal Kumar Bhatia, "Design of a Least Cost Vertical Search Engine based on DSHWC" IJIRR, accepted and will be processed in Vol 7,Issue 1, 2016.

[76] Jyoti Gautam, Ela Kumar, and Mehjabin Khatoon Semantic Web Improved with IDF Feature of the TFIDF Algorithm in Proceedings of the International Multi Conference of Engineers and Computer Scientists 2014 Vol I, IMECS 2014, March 12 - 14, 2014, Hong Kong.

[77] Sonali Gupta, Komal Kumar Bhatia: A Comparative Study of Hidden Web Crawlers in International Journal of Computer Trends and Technology (IJCTT) – volume 12 number 3 – Jun 2014, pp 111- 118, ISSN: 2231-2803

[78] Sonali Gupta, Komal Kumar Bhatia: Hidden Web Resource Discovery through semantic understanding of Search Form interfaces in IEEE international Conference on Advanced Computing and Communication technologies, ICACCT-2014 proceedings by IEEE Xplore.

[79] Sini Shibu, Aishwarya Vishwakarma and Niket Bhargava, "A combination approach for Web Page Classification using Page Rank and Feature Selection Technique" , International Journal of Computer Theory and Engineering, Vol.2, No.6, December, 2010

[80] H. He, W. Meng, C. T. Yu and Z. Wu, "Automatic extraction of web search interfaces for interface schema integration.," in WWW (Alternate Track Papers & Posters), ACM, 2004, pp. 414-415.

[81] Ping Wu, J.-R. Wen, H. Liu, and W.-Y. Ma. Query Selection Techniques for Efficient Crawling of Structured Web Sources. In ICDE, 2006.

[82] Manuel Álvarez, Juan Raposo, Alberto Pan, Fidel Cacheda, Fernando Bellas, Víctor Carneiro: Crawling the Content Hidden Behind Web Forms. In Proceedings of the 2007 International conference on Computational Science and its applications, Published by Springer-Verlag Berlin, Heidelberg, 2007

[83] De Carvalho Fontes and F. Soares Silva, "SmartCrawl: A New Strategy for the Exploration of the Hidden Web", ACM Transaction on WIDM'04, Washington, DC, USA, November 2004.

[84] Saeko Nomura, Satoshi Oyama, Tetsuo, Analysis and Improvement of HITS Algorithm for Detecting Web Communities[C]. Proceeding of the Symposium on the Applications and the Internet (SAINT'02),2002.

[85] Amit C. Awekar, Jaewoo Kang. Selective Approach To Handing Topic Oriented Tasks On The World Wide Web[C].Proceeding of the 2007 IEEE Symposium on Computational intelligence and Data Ming(CIDM2007),2007:343-348.

[86] www.eecis.udel.edu

[87] Gary William Flake, Steve Lawrence, C.Lee Giles. Efficient Identification of Web Communities[C].Proceedings of the sixth ACM SIGKDD international conference on knowledge discovery and data mining.2000:150-160.

[88] Chau M, Chen H C. A machine learning approach to web page filtering using content and structure analysis[J]. Decision Support Systems. ELSEVIER 2008:482-494.

[89] Xing Wenpu and Ghorbani Ali. Weighted PageRank Algorithm[C]. Proceedings of the Second Annual Conference on Communication Networks and Services Research(CNSR'04),2004.

[90] Zheng Chen, Li Tao, Wang Jidong, Liu Wenyin, Ma Wei-Ying. A Unified Framework for Web Link Analysis[C]. Proceedings of the 3rd InternationalConference on Web Information Systems Engineering (WISE'02).2002IEEE.

[91] Xu Yinghui, Umemura Kyoji. Literal-Matching-Biased Link Analysis[J].S.H.Myaeng et al.(Eds.): AIRS 2004, LNCS 3411,Springer-Verlag Berlin Heidelberg 2005:153-164.

[92] Soumen Chakrabarti, Byron E.Dom David Gibson. Mining the Link Structure of the World Wide Web[C].IEEE Computer 1999,32(8)

[93] Boris Plejic, Branislav Vujnovic,Roberto Penco: Transforming unstructured data from Scattered Sources into Knowledge,IEEE 2008

[94] Nan Du,Hong Peng,Wenfeng Zhang: Application of Modified Gentic Algorithm in feature extraction of the Unstructured Data. IEEE 2008

[95] K. C.-C. Chang, B. He, C. Li, M. Patel, and Z. Zhang. Structured Databases on the Web: Observations and Implications. SIGMOD Record, 33(3):61–70, 2004.

[96] K. C.-C. Chang, B. He, and Z. Zhang. Toward Large-Scale Integration: Building a Meta Querier over Databases on the Web. In Proc. of CIDR, pages 44–55, 2005.

[97] J. Cope, N. Craswell, and D. Hawking. Automated Discovery of Search Interfaces on the Web. In Proc. of ADC, pages 181–189, 2003.

[98] Susanne buklen, prdrojosemarron, serenafritish ,User centric Walk:An integrated approach for modeling the browsing behavior of users on the web proceeding of the 38th annual simulation symposium. IEEE 2005

[99] Douglas A. Drucken miller: The organizing pattern of Interaction: a knowledge centric approach IEEE 2008.

[100] Kesun and Fergshanbai Mining weighted association rules without pre assigned weights IEEE transaction on knowledge and data engineering,vol20 No. 4 April 2008

[101] Olfanasraoui, A web usage mining frame work for mining evolving user profile in dynamic web site .IEEE Transaction Vol20 No. 2 Feb 2008

[102] Robert Cooley, Bamshad Mobasher, Jaideep Srivastava (1997). Web Mining: information and pattern discovery on the World Wide Web. Proc.ninth IEEE int't conf. Tolls with AI(ICTAI'97_PP.558-567 1997

[103] Robert Cooley, Bamshad Mobasher, Jaideep Srivastava, Data preparation for mining World Wide Web browsing patterns 1999

[104] Cooley R., Mobasher B.,and Srivastava J., Data preparation for mining World Wide Web browsing patterns , Knowledge and Information Systems, V1(1), 1999

[105] Jiawei Han, Man Xin, OsmarZaiane, Discovering web access patterns and trends byapplying OLAP and data mining technology on web logs, In Proceedings of the Fifth IEEE Forum on Research and Technology Advances in Digital Libraries, 1998.

[106] Bamshad Mobasher, Namit Jain, Eui-Hong Han, Jaideep Srivastava . Web mining: pattern discovery from World Wide Web Transactions 1996.

[107] Robert Cooley, Bamshad Mobasher, Jaideep Srivastava, Grouping Web page references into transactions for mining World Wide Web browsing patterns 1997.

[108] Christain von der Weth,Anwitaman Datta: Multiterm Keyword Search in NoSQL System,IEEE Computer Society,2012

[109] http://www.informationretrieval.org/ online edition(c) 2009 Cambridge UP.

[110] Daniel Rocco, James Caverlee, Ling Liu, and Terence Critchlow ,Domain-specific Web Service Discovery with Service Class Descriptions: International Conference on Web Services Orlando, FL, United States,July 12, 2005.

APPENDIX-A

SEARCH LOG

The following is the list of search log id with email and Book id.

Searchlogid	Email	Bookid
1	abcd@gmail.com	2
2	abcd@gmail.com	3
3	abcd@gmail.com	4
4	abcd@gmail.com	5
5	abcd@gmail.com	6
6	abcd@gmail.com	7
7	abcd@gmail.com	8
8	abcd@gmail.com	2
9	abcd@gmail.com	3
10	abcd@gmail.com	4
11	abcd@gmail.com	5
12	abcd@gmail.com	6
13	abcd@gmail.com	7
14	abcd@gmail.com	8
15	sudhakarayush@gmail.com	2
16	sudhakarayush@gmail.com	3
17	sudhakarayush@gmail.com	4
18	sudhakarayush@gmail.com	5
19	sudhakarayush@gmail.com	6
20	sudhakarayush@gmail.com	7
21	sudhakarayush@gmail.com	8
22	sudhakarayush@gmail.com	2
23	sudhakarayush@gmail.com	3
24	sudhakarayush@gmail.com	4
25	sudhakarayush@gmail.com	5
26	sudhakarayush@gmail.com	6
27	sudhakarayush@gmail.com	7
28	sudhakarayush@gmail.com	8
29	sudhakarayush@gmail.com	6
30	sudhakarayush@gmail.com	14
31	sudhakarayush@gmail.com	15
32	sudhakarayush@gmail.com	16
33	sudhakarayush@gmail.com	2
34	sudhakarayush@gmail.com	3

Searchlogid	Email	Bookid
35	sudhakarayush@gmail.com	4
36	sudhakarayush@gmail.com	5
37	sudhakarayush@gmail.com	6
38	sudhakarayush@gmail.com	7
39	sudhakarayush@gmail.com	8
40	sudhakarayush@gmail.com	1
41	sudhakarranjan@gmail.com	2
42	sudhakarranjan@gmail.com	3
43	sudhakarranjan@gmail.com	4
44	sudhakarranjan@gmail.com	5
45	sudhakarranjan@gmail.com	6
46	sudhakarranjan@gmail.com	7
47	sudhakarranjan@gmail.com	8
48	sudhakarranjan@gmail.com	2
49	sudhakarranjan@gmail.com	3
50	sudhakarranjan@gmail.com	4
51	sudhakarranjan@gmail.com	5
52	sudhakarranjan@gmail.com	6
53	sudhakarranjan@gmail.com	7
54	sudhakarranjan@gmail.com	8
55	sudhakarranjan@gmail.com	2
56	sudhakarranjan@gmail.com	3
57	sudhakarranjan@gmail.com	4
58	sudhakarranjan@gmail.com	5
59	sudhakarranjan@gmail.com	6
60	sudhakarranjan@gmail.com	7
61	sudhakarranjan@gmail.com	8
62	sudhakarranjan@gmail.com	14
63	sudhakarranjan@gmail.com	15
64	sudhakarranjan@gmail.com	16
65	sudhakarranjan@gmail.com	2
66	sudhakarranjan@gmail.com	3
67	sudhakarranjan@gmail.com	4
68	sudhakarranjan@gmail.com	5
69	sudhakarranjan@gmail.com	6
70	sudhakarranjan@gmail.com	7
71	sudhakarranjan@gmail.com	8
72	sudhakarranjan@gmail.com	2
73	sudhakarranjan@gmail.com	3
74	sudhakarranjan@gmail.com	4
75	sudhakarranjan@gmail.com	5

Searchlogid	Email	Bookid
76	sudhakarranjan@gmail.com	6
77	sudhakarranjan@gmail.com	7
78	sudhakarranjan@gmail.com	8
79	sudhakarranjan@gmail.com	11
80	sudhakarranjan@gmail.com	12
81	sudhakarranjan@gmail.com	2
82	sudhakarranjan@gmail.com	3
83	sudhakarranjan@gmail.com	4
84	sudhakarranjan@gmail.com	5
85	sudhakarranjan@gmail.com	6
86	sudhakarranjan@gmail.com	7
87	sudhakarranjan@gmail.com	8
88	sudhakarranjan@gmail.com	14
89	sudhakarranjan@gmail.com	15
90	sudhakarranjan@gmail.com	16
91	sudhakarranjan@gmail.com	2
92	sudhakarranjan@gmail.com	3
93	sudhakarranjan@gmail.com	4
94	sudhakarranjan@gmail.com	5
95	sudhakarranjan@gmail.com	6
96	sudhakarranjan@gmail.com	7
97	sudhakarranjan@gmail.com	8
98	sudhakarranjan@gmail.com	19
99	sudhakarranjan@gmail.com	9
100	sudhakarranjan@gmail.com	14
101	sudhakarranjan@gmail.com	15
102	sudhakarranjan@gmail.com	16
103	sudhakarranjan@gmail.com	17
104	sudhakarranjan@gmail.com	14
105	sudhakarranjan@gmail.com	15
106	sudhakarranjan@gmail.com	16
107	sudhakarranjan@gmail.com	19
108	sudhakarranjan@gmail.com	14
109	sudhakarranjan@gmail.com	15
110	sudhakarranjan@gmail.com	16
111	sudhakarranjan@gmail.com	14
112	sudhakarranjan@gmail.com	15
113	sudhakarranjan@gmail.com	16
114	sudhakarranjan@gmail.com	14
115	sudhakarranjan@gmail.com	15
116	sudhakarranjan@gmail.com	16

Searchlogid	Email	Bookid
117	sudhakarranjan@gmail.com	14
118	sudhakarranjan@gmail.com	15
119	sudhakarranjan@gmail.com	16
120	sudhakarranjan@gmail.com	2
121	sudhakarranjan@gmail.com	3
122	sudhakarranjan@gmail.com	4
123	sudhakarranjan@gmail.com	5
124	sudhakarranjan@gmail.com	6
125	sudhakarranjan@gmail.com	7
126	sudhakarranjan@gmail.com	8
127	sudhakarranjan@gmail.com	2
128	sudhakarranjan@gmail.com	3
129	sudhakarranjan@gmail.com	4
130	sudhakarranjan@gmail.com	5
131	sudhakarranjan@gmail.com	6
132	sudhakarranjan@gmail.com	7
133	sudhakarranjan@gmail.com	8
134	sudhakarranjan@gmail.com	2
135	sudhakarranjan@gmail.com	3
136	sudhakarranjan@gmail.com	4
137	sudhakarranjan@gmail.com	5
138	sudhakarranjan@gmail.com	6
139	sudhakarranjan@gmail.com	7
140	sudhakarranjan@gmail.com	8
141	s@gmail.com	2
142	s@gmail.com	3
143	s@gmail.com	4
144	s@gmail.com	5
145	s@gmail.com	6
146	s@gmail.com	7
147	s@gmail.com	8
148	s@gmail.com	2
149	s@gmail.com	3
150	s@gmail.com	4
151	s@gmail.com	5
152	s@gmail.com	6
153	s@gmail.com	7
154	s@gmail.com	8
155	s@gmail.com	2
156	s@gmail.com	3
157	s@gmail.com	4

Searchlogid	Email	Bookid
158	s@gmail.com	5
159	s@gmail.com	6
160	s@gmail.com	7
161	s@gmail.com	8
162	s@gmail.com	2
163	s@gmail.com	3
164	s@gmail.com	4
165	s@gmail.com	5
166	s@gmail.com	6
167	s@gmail.com	7
168	s@gmail.com	8
169	s@gmail.com	13
170	s@gmail.com	17
171	s@gmail.com	18
172	s@gmail.com	2
173	s@gmail.com	3
174	s@gmail.com	4
175	s@gmail.com	5
176	s@gmail.com	6
177	s@gmail.com	7
178	s@gmail.com	8
179	s@gmail.com	3
180	s@gmail.com	5
181	s@gmail.com	8
182	s@gmail.com	2
183	s@gmail.com	3
184	s@gmail.com	4
185	s@gmail.com	5
186	s@gmail.com	6
187	s@gmail.com	7
188	s@gmail.com	8
189	s@gmail.com	2
190	s@gmail.com	3
191	s@gmail.com	4
192	s@gmail.com	5
193	s@gmail.com	6
194	s@gmail.com	7
195	s@gmail.com	8
196	s1@gmail.com	13
197	s1@gmail.com	17
198	s1@gmail.com	18

Searchlogid	Email	Bookid
199	s@gmail.com	9
200	s@gmail.com	10
201	s@gmail.com	9
202	s@gmail.com	10
203	s@gmail.com	14
204	s@gmail.com	15
205	s@gmail.com	16
206	s@gmail.com	2
207	s@gmail.com	3
208	s@gmail.com	4
209	s@gmail.com	5
210	s@gmail.com	6
211	s@gmail.com	7
212	s@gmail.com	8
213	s@gmail.com	2
214	s@gmail.com	3
215	s@gmail.com	4
216	s@gmail.com	5
217	s@gmail.com	6
218	s@gmail.com	7
219	s@gmail.com	8
220	s@gmail.com	14
221	s@gmail.com	15
222	s@gmail.com	16
223	s@gmail.com	2
224	s@gmail.com	3
225	s@gmail.com	4
226	s@gmail.com	5
227	s@gmail.com	6
228	s@gmail.com	7
229	s@gmail.com	8
230	s@gmail.com	13
231	s@gmail.com	17
232	s@gmail.com	18
233	s@gmail.com	13
234	s@gmail.com	17
235	s@gmail.com	18
236	s@gmail.com	2
237	s@gmail.com	3
238	s@gmail.com	4
239	s@gmail.com	5

Searchlogid	Email	Bookid
240	s@gmail.com	6
241	s@gmail.com	7
242	s@gmail.com	8
243	s@gmail.com	9
244	s@gmail.com	10
245	s@gmail.com	13
246	s@gmail.com	17
247	s@gmail.com	18
248	s@gmail.com	2

APPENDIX-B

BUY LOG

The following is list of Buy Log with Buy log id, email and Bookid.

Buylogid	Email	Bookid
1	sudhakarranjan@gmail.com	3
2	sudhakarranjan@gmail.com	2
3	sudhakarranjan@gmail.com	3
4	sudhakarranjan@gmail.com	2
5	s1@gmail.com	13
6	s@gmail.com	9
7	s@gmail.com	7
8	s@gmail.com	14

APPENDIX-C

USER LOG

The following is the list of uid along with user name, email and password.

uid	Username	Email	Password
1	Test name	test@abc.com	Abc
2	Abcd	abcd@gmail.com	Abcd
3	Sudhakar	sudhakarayush@gmail.com	Ayush
4	Sudhakar Ranjan	sudhakarranjan@gmail.com	Ayush
5	Sudhakar	sudhakar@gmail.com	Aba
6	Su	s@gmail.com	Abc
7	s1@gmail.com	s1@gmail.com	Abc

BRIEF PROFILE OF RESEARCH SCHOLAR

Sudhakar Ranjan is a PhD scholar in Computer Engineering Department of YMCA University of Science & Technology, Faridabad. He has completed his B.E from Nagpur University in Computer Technology in 1997. In 2006, he completed M.Tech., in Computer Engineering from YMCAIE, Faridabad. He has more than 18 years of industry and teaching experience. Presently, he is working as a coordinator in CSE Department of Apeejay Stya University, Sohna Gurgaon

LIST OF PUBLICATION

List of published paper in International Journal

S.No	Title of the paper along with volume, Issue No, year of publication	Publisher	Impact Factor	Refereed / Non-Refereed	Whether you paid any money or not for publication	Remarks
1.	Design of a Least Cost Vertical Search Engine based on DSHWC, in Vol VII, Issue II,IJIRR 2017.	IJIRR		Refereed	No	**Indexed by DBLP,** Google Scholar, IJIRR
2.	Query Interface Integrator for Domain Specific Hidden Web Vol IV, Issue I/III, Oct.-Dec. 2013	International Journal of Computer Engineering & Applications	Impact Factor: 2.84	Refereed	yes	**Indexed by DBLP,** Google Scholar, arxiv.org
3.	Indexing for Vertical Search Engine: Cost Sensitive" Volume 3, Issue 10, October 2013.	International Journal of Emerging Technology & Advanced Engineering	Impact Factor 2.324	Refereed	yes	Google ,Yahoo, Entire Web
4.	Indexing for Domain Specific Hidden Web Vol VII, Issue I, July 2014	International Journal of Computer Engineering & Applications	Impact Factor: 2.84	Refereed	No	Google Scholar
5.	Web Log for Domain Specific Hidden, Vol VII, Issue I, July 2014	International Journal of Computer Engineering & Applications	Impact Factor: 2.84	Refereed	yes	Google Scholar
6.	Transaction in Hidden Web,Vol VIII, Issue II, November 2014	International Journal of Computer Engineering & Applications	Impact Factor: 2.84	Refereed	yes	Google Scholar