Comparative Analysis of the Information Retrieval Strategies in Web Crawling

Thesis submitted in partial fulfillment of the requirements for the award of degree of

Master of Engineering

in

Information Security

Submitted By:

Chandni Saini

(Roll No. 801433007)

Under the supervision of:

Mr. Vinay Arora

(Assistant Professor)

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT

THAPAR UNIVERSITY

PATIALA – 14700

June 2016

Certificate

I hereby certify that the work which is being presented in the thesis entitled, "***Comparative Analysis of the Information Retrieval Strategies in Web Crawling***", in partial fulfillment of the requirements for the award of degree of Master of Engineering in *Information Security* submitted in Computer Science and Engineering Department of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of *Mr. Vinay Arora* and refers other researcher's work which are duly listed in the reference section.

The matter presented in the thesis has not been submitted for award of any other degree of this or any other University.

(Chandni Saini)

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

(Mr. Vinay Arora)
Assistant Professor
CSED, Thapar University
Patiala

Countersigned by

(Dr. Maninder Singh)
Head of Department
CSED, Thapar University
Patiala

(Dr. S. S. Bhatia)
Dean (Academic Affairs)
Thapar University
Patiala

Acknowledgement

First of all I would like to thank the Almighty, who has always guided me to work on the right path of the life. It is a great privilege to express my gratitude and admiration towards my respected supervisor **Mr. Vinay Arora** Assistant Professor, Computer Science and Engineering Department, Thapar University, Patiala. He has been an esteemed guide and great support behind achieving this task. This work would not have been possible without the encouragement and able guidance of him. I also thank my supervisor for his time, patience, discussions and valuable comments. His enthusiasm and optimism made this experience both rewarding and enjoyable. I am truly grateful to him for extending his total co-operation and understanding whenever I needed help and guidance from him. I am also heartily thankful to **Dr. Maninder Singh**, Associate Professor and Head, Computer Science & Engineering Department and **Dr. Jhilik Bhattacharya**, PG coordinator, for motivation and providing uncanny guidance and support throughout the preparation of the thesis report.

I will be failing in my duty if I do not express my gratitude to **Dr. S. S. Bhatia**, Senior Professor and Dean of Academic Affairs, for making provisions of infrastructure such as library facilities, computer labs equipped with net facilities, immensely useful for the learners to equip themselves with the latest in the field.

I am also thankful to the entire faculty and staff members of Computer Science and Engineering Department for their direct-indirect help, cooperation, love and affection, which made my stay at Thapar University memorable. Last but not least, I would like to thank my family for their wonderful love and encouragement, without their blessings none of this would have been possible.

Chandni Saini
(801433007)

Abstract

In today's scenario, World Wide Web (WWW) is flooded with huge amount of information. Due to growing popularity of the internet, finding the meaningful information among billions of information resources on the WWW is a challenging task. The Information Retrieval (IR) provides relevant information to the end users which satisfy their requirement. Search engine is used to extract valuable information from the internet. Web crawler is the principal part of search engine; it is an automatic script or program which can browse the web in automatic manner. This process is known as web crawling. In this literature, review on the strategies of information retrieval in web crawling has been presented that are classified into four main categories *viz:* focused, distributed, incremental and hidden web crawler. Finally, on the basis of user customized parameters the comparative analysis of various IR strategies has been performed.

Table of Contents

Certificate…… **i**

Acknowledgement…….. **ii**

Abstract…… **iii**

Table of Contents…….. **iv**

List of Figures…… **vi**

List of Tables…….. **viii**

Chapter 1 Introduction…… **1**

1.1World Wide Web (WWW)…….. 1

1.2 Introduction to internet and its working…….. 2

1.3 Introduction to search engine and its working…….. 4

1.4 Information retrieval and its categories…….. 6

1.5 Web crawler…….. 13

1.6 Introduction to web crawling…….. 13

1.7 Working of web crawler…….. 14

1.8 Components of the web crawler…….. 15

1.9 Web crawler features…….. 16

1.10 Types of web crawler…….. 17

1.10.1 Focused web crawler…….. 17

1.10.2 Distributed web crawler…….. 18

1.10.3 Incremental Web Crawler…….. 20

1.10.4 Hidden Web Crawler…….. 21

Chapter 2 Literature Survey…….. 23

2.1 Strategies of information retrieval in web crawling…….. 23

2.1.1 Focused web crawler…… 24

2.1.2 Distributed web crawler…….. 30

2.1.3 Incremental web crawler…….. 38

2.1.4 Hidden web crawler…….. 43

Chapter 3 Problem Statement…….. 50

Chapter 4 Methodology…….. 51

Chapter 5 Experimental Results…….. 52

5.1 Tools……..... 52

5.2 Implementation…… 57

Chapter 7 Conclusion and Future Scope…….... 62

References……..……..... 65

Annexure

I List of Publications……..…..... 75

II Video Link……..... 76

III Plagiarism Certificate……..... 77

List of Figures

Figure 1.1: Key layers of the internet……..... 1

Figure 1.2: Internet users in the world by regions [November 2015]…… 2

Figure 1.3: Internet framework…….. 3

Figure 1.4: Working of the internet…… 4

Figure 1.5: Several types of search engines have used in the world…… 4

Figure 1.6: Working of the search engine……..... 5

Figure 1.7: A general framework for information retrieval……. 6

Figure 1.8: Categorization of information retrieval approach ……..…… 7

Figure 1.9: Information pyramid…….. 12

Figure 1.10: Machine learning……..... 12

Figure 1.11: Web crawler…… 13

Figure 1.12: Working of a web crawler…….. 14

Figure 1.13: Components of the web crawler……. 15

Figure 1.14: Architecture of the incremental crawler……. 20

Figure 1.15: Hidden web……. 21

Figure 2.1: Strategies of information retrieval in web crawling ……. 23

Figure 2.2 Architecture of IgooG……. 34

Figure 2.3 Architecture of self adjusting refresh time based incremental crawler… 41

Figure 5.1: Downloading of URLs using Nutch source version in eclipse…….. 52

Figure 5.2: : Various hadoop jars used by the Nutch 1.9……. 53

Figure 5.3: Solr giving results to queries passed on by nutch……. 54

Figure 5.4: Java 1.8.0_31……. 54

Figure 5.5: View of Eclipse IDE……. 55

Figure 5.6: Progress of building nutch project using Apache Ant…… 55

Figure 5.7: IvyDe2.2.0…….. 56

Figure 5.8: Subclipse jars downloaded from eclipse marketplace…… 56

Figure 5.9: Maven m2e plugins being used in the eclipse IDE…… 57

Figure 5.10: Cygwin64……. 57

Figure 6.1: Comparison of methodologies of information retrieval in web crawling……. 63

List of Tables

Table 5.1: Number of URLs crawled by focused and distributed …….58

Table 5.2: Comparison of the total number of iterations in focused and distributed web crawler……..59

Table 5.3: Comparison of the time taken in seconds in focused and distributed web crawler……..60

Table 5.4: Comparison of the number of hosts and type of threading in focused and distributed web crawler…….61

Chapter 1

Introduction

1.1 World Wide Web (WWW)

The internet is a global system to make interconnection between computer networks. On the other hand, the World Wide Web (WWW) is a collection of global text documents and resources which have been linked by the Uniform Resource Locators (URIs) and hyperlinks and thus can be accessed through remote servers with the use of internet.

WWW can be defined as the huge space of information and data where web resources have been identified by its unique URLs, and hypertext links have been used for interlinking between the web pages for accessing on the internet. WWW has been developed in 1989 by Tim Berners-Lee, an English Scientist. Tim had termed the initial web browser in 1990 when he was an employee of French Conseil Européen pour la Recherche Nucléaire (CERN) in Switzerland.

At the beginning, it was known as web, and then WWW has become the central development of information aging and now, it is an important tool for billions people who used it to interact over internet [1].

Fig.1.1 shows the key layers of the internet. The WWW layer is placed at top of the internet layer which helps people in understanding the functional aspects of the internet as shown in the fig.1.1.

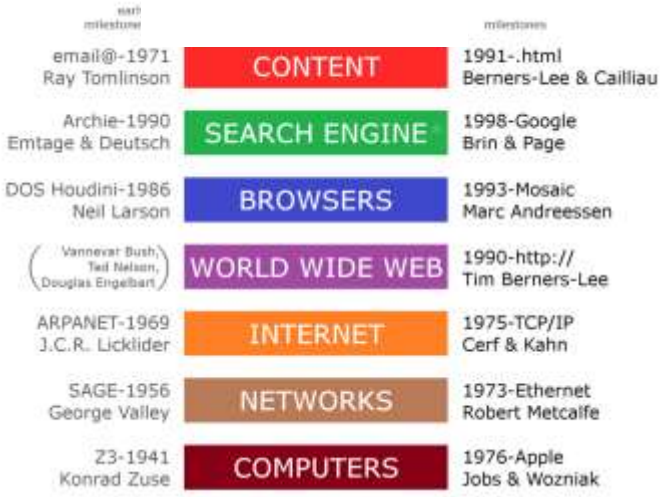


Fig.1.1: Key layers of the internet [1]

1.2 Introduction to internet and its working

All computers of world are linked-up by a virtual network or mesh known as internet. There's only your computer system on desk and on the other end, person has his own system to communicate with you. But in the mid of these two communication systems there are probably near about a dozen processes and computer systems bridging the gap. The fig.1.2 displays the internet users in the world by regions [November 2015].

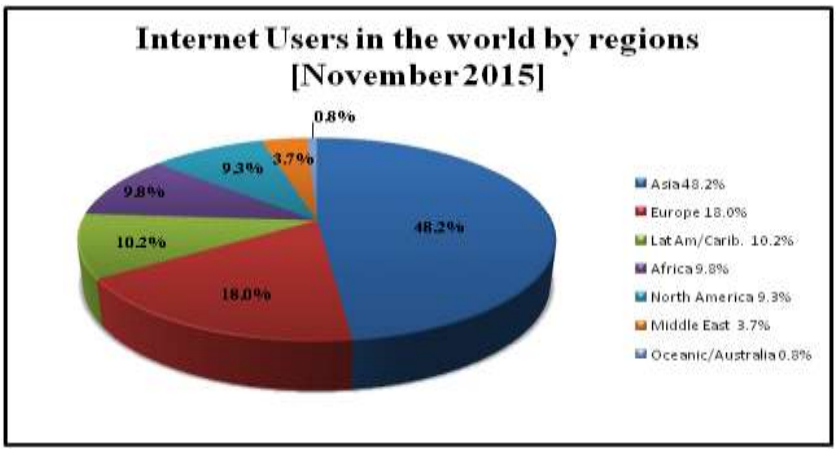


Fig.1.2: Internet users in the world by regions [November 2015]

The internet is a combination of smaller networks over the world which in turn is combined together into a global network. This global network contains billions of public, private, business, academic and governmental networks. The internet is used to connect the millions of computers which are known as hosts. In internet, we use Transmission Control Protocol/Internet Protocol (TCP/IP) communication protocol over internet. These computer systems over internet have been interlinked via various communication media devices. Some common communication device and media used are fiber optic cables, telephone lines, satellite and microwave etc. Fig.1.3 displays the internet framework which consists of Internet Service Provider (ISP), ISP's web server, modem, ISP's network and telephone line *etc*.

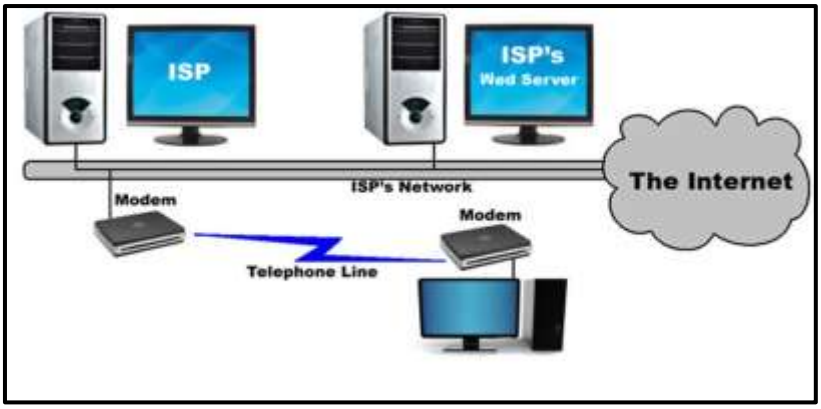


Fig.1.3: Internet framework [2]

In the world of interconnected global internet, anyone can communicate easily with all others who are connected in the network or they can text or call to each other. In around 20 years internet has been extended to around different 210 nations.

There is no specific organization or company which controls the global internet. This is held by different government agencies, private companies, universities, research organizations *etc*. which are interconnected or interlinked with each other. Moreover, we can understand that internet is combination of millions of devices all over the world linked together. The following fig.1.4 displays how the internet works.

With the use of phone-line modem, a personal computer is linked to internet with the help of cable modem and Digital Subscriber Line (DSL). The modem can be used to communicate with server over the Internet Server Provider (ISP). ISP is an organization which offers internet connection to customers. Various ISP companies are available in every country over the world. Any customer can take an internet from ISP companies to connect with internet.

The computer system in university or business organization can connect with Local Area Network (LAN), with the use of Local Area Network Card (LAN Card) or Network Interface Card (NIC). The LAN card of a company or a university is connected to ISP server using higher-speed phone line such as TI Line. The TI Line can handle 1.5 million approximately bits per second. The device is connected to ISP server which is further connected to the huge ISP. The huge internet server provider has maintained fiber-optic lines under satellite or cables links. In this manner, each device is connected with internet and other device which is also a part of this internet [2].

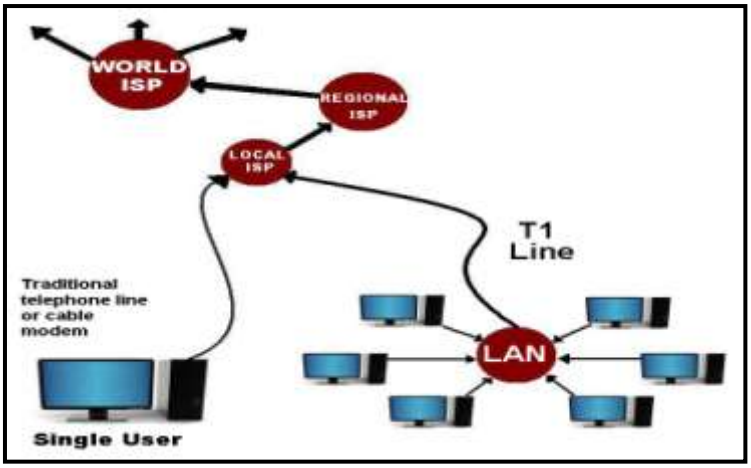


Fig.1.4: Working of the internet [2]

1.3 Introduction to search engine and its working

The search engine is script or software program which can be accessed through internet to find the files or documents using keywords provided by the user, returning result files most relevant to the keyword. Now-a-days, thousands of search engines are available over the internet and each search engine has its different features and abilities. The first search engine was developed called Archie and it has been used to find the File Transfer Protocol (FTP) files. The first text-based search engine developed was called as Veronica. Now-a-days, the world's most famous and popular search engine is the "Google". Fig.1.5 displays several types of search engines that are prevalent in the world.

Fig.1.5: Several types of search engines have used in the world [3]

Web search engines send their spider to fetch as many documents as possible. One more software known as indexer is an important part of search engine. It can read the files and documents followed by indexing based on word contained in every document. Every search engine can use the proprietary algorithm to create indices to return ideal meaningful results for every query.

Common search engine types:

Out of the various types of web search engines, the most common types including the following:-

1. Local search engine: It has been created to use for Compact Disk Read Only Memory (CDROM), offline Personal Computer (PC) or search usage in local area network.

2. Meta search engine: This search engine takes the queries of other search engines and then combines the conclusions that are received from all.

3. Blog search engine: This search engine can be used for blogosphere. The blog search engine provides search results indexed from blogs.

Fig.1.6 displays the functional architecture of search engine.

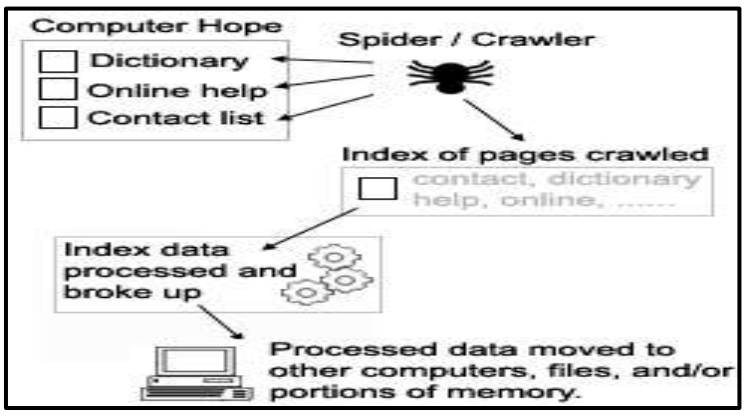


Fig.1.6: Working of the search engine [3]

As discussed in this figure, a crawler or spider is a source of all search engine data that automatically visited pages and indexes their contents.

When the page has been crawled the data that contained within the page is processed, which involves these steps given below.

- Record links with other pages.

- Strip out to stop the words.
- Store the remaining words in the page and frequency they occur.
- Store the information of embedded media or images.

The collection of data can be used for ranking the page and is main technique for search engine which is used for determine if page will be displayed and in what order [3].

1.4 Information retrieval and its categories

A technique used to store and recover the information is called Information Retrieval (IR) which is used to record the data with the help of computer system.

IR is an activity to obtain the information from resources which is relevant the needed information from number of informational resources. Search is based upon full-text indexing.

The information retrieval was termed in 1951 by Calvin Mooers.

This process starts when user is entering a query in the system. These queries have simple statements for required search information strings in the search engine. In IR query which is not uniquely identified with an object in collection. Moreover, various objects can be matched with queries on different relevancy degree.

The aim of information retrieval is to give the necessary information to users which satisfy their need. Along with the use "document" word format in general term which includes multimedia objects and non-textual data.

The Retrieval Process:

The information retrieval system has copied with three individual processes *viz*: represent the documents, represent the needed user information and compare the representations of both. Fig.1.7 displays a general framework for information retrieval.

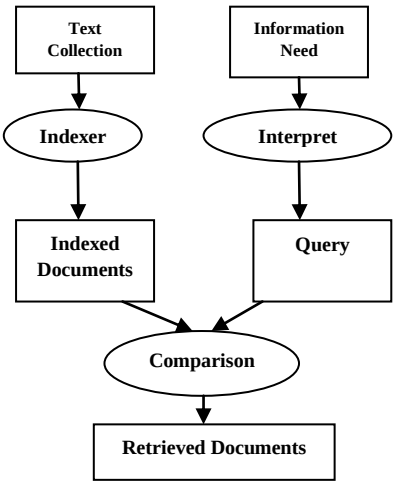


Fig.1.7: A general framework for information retrieval

Information retrieval categories:

Information retrieval is of two types: traditional information retrieval and automated information retrieval, which are further divided into various sub-types. Fig.1.8 shows the categorization of information retrieval approaches.

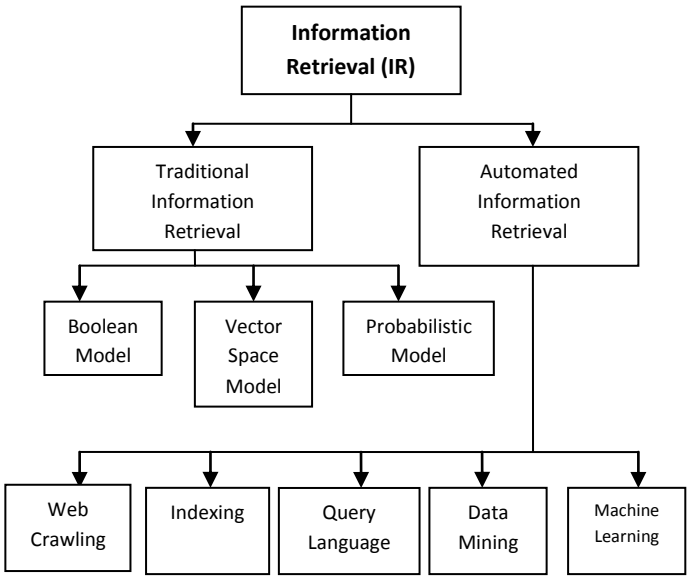


Fig.1.8: Categorization of information retrieval approaches

a) Traditional information retrieval

These systems represent most related information to searcher by putting masks on them. They represent specific information for searching but have no knowledge of others' data or their existence. This task is achieved as a central result or includes the feedback loop where customer can refine the query.

In these systems not only customer comes with limited subset of the documents but also the relationships among documents are not presented clearly.

Traditional information retrieval models:-

The traditional IR models were developed to retrieve the data or information: Vector Space Model, Boolean Model and Probabilistic Model.

1. Boolean model

It is based upon the Boolean algebra which is named after English mathematician George Boole (1815-1864). Boolean algebra plays an important role in IR systems and has more influence in computer science. Boolean retrieval can consider one among the three very crucial models for IR systems, two being the probabilistic model and vector model.

This was the first model to be used for information retrieval and is displayed as the most criticized model. Moreover, model considers a query term which is an unambiguous definition of a number of informational documents. Boole has described the three operators, logical sum named OR, logical product known as AND and logical difference known as NOT.

It has the following strengths which are given below:

a) This is simpler and easier to computationally implement and efficient. However, this is a standard model for operational retrieval systems and current large-scale and various major online data services use it.

b) This model possesses an expressive and clear system power. It requires an efficient query for an unambiguous and exhaustive selection.

c) It provides a multitude technique for narrow or broad a query.

d) It is especially effective in further stages to search the processes because of the correctness and clarity with which relationship between concepts has been displayed [4].

2. Vector space model

It is an algebraic model to represent the text documents as vectors identifiers just like indexing terms. This is employed for retrieving information, info filter and indexing or relevant rankings. Examples of IR systems are: SMART, Lucene, Lemur, JFeret, Inquire etc. It was first used in SMART IR System. Vector Space Model has natural language documents which is represented in formal manner by using vectors in the multi-dimensional space. It was developed in 1960's by Gerard Salton and used in SMART system.

Salton's classic weight is provided with equation, which is given below:

Term Weight =

where

- tf_i = number of term i occurred in document or term frequency.
- df_i = number of total documents containing term i or document frequency

- D = number of database documents.

Various models which are extracting vectors model from queries and documents have been taken from equation.

This model has various advantages over Boolean model, which are:

a) Model based upon linear algebra.
b) Weights terms are not binary.
c) Allows for computing the continue degree between documents and queries.
d) According to relevancy allows ranking documents.
e) Partial matching is allowed.

This model has following limitations:

a) Longer docs are represented as poor because they have similar poor values.
b) Search the keywords which are precisely matching terms document which words are substrings may result in "false positive matching".

3. Probabilistic model

These models work for document retrieval as probabilistic inference. Their similarities are calculated as probabilities to identify document that can be relevant for provided query. Their theorems work same as Bayes' theorem which have often used in models.

This model is based upon the principle of probability ranking in which an IR is supposed to rank the documents according to the probability of relevance to query, provided there is available evidence. This principle can be taken into account when there is uncertainty in representation of needed documents and information. There are a number of evident sources which can be used by probabilistic retrieval methods. The one commonly used is statistical distribution in both non-relevant and relevant documents.

Probabilistic model has some strengths, which are:

a) It provides to user with relevant ranks of retrieval documents. Therefore, enabling the users to overcome the output to set relevant threshold or specify specific document is required.

b) Their Query is easy for formulation because user cannot learn natural language and queries language.

c) The inherent uncertainty in query concept is represented.

This model has following shortcomings:

a) It has expressed limited power. For example: NOT operation is not used in case of when positive weights have been used. It has been proved that only 2N*N of the 22N possible boolean queries are generated by statistical methods which use linear weighted sums to rank documents [5].

b) Automated information retrieval

These systems are used to decrease "information overload". Various public libraries and universities use information retrieval systems for providing an access to journals, books and informational documents. Web search engines have many visible information retrieval applications. This is a portion of large image and vital tool used in searching, filtering and browsing.

The computer-based IR systems developed initially had limited capabilities comparing to today's computer systems. According to multiple computer programs, information retrieval systems have been modeled in several paths after traditional search systems, particularly for card catalogue and the user. This type of IR is used in various fields such as: web crawling, indexing, query language, data mining and machine learning.

Automatic information retrieval is used in various fields:

1. Web crawler

It is an automatic script which can download the content from the WWW in an automatic manner. The "web crawler" systematically crawls pages and looks for keyword contained in pages and the anchor texts all, returning the info to server search engine for index.

2. Indexing

Indexing is an important portion of IR system. Indexing is the process of extraction rather than content analysis.

3. Query Language

Query language is used to inject query into the database where the semantic query cannot be defined by precisely rendering the syntax but by interpreting the query results.

Information Retrieval (IR) Queries:

Boolean queries:-

Keywords are combined with boolean operators such as OR: (e_1 OR e_2), BUT: (e_1 BUT e_2) Satisfies e_1 but not e_2 and AND: (e_1 AND e_2). There is negation allowed with the use of BUT that allows use of inverted index for efficient filtering of retrieved set. A naïve user faces with Boolean logic.

"Natural Language" queries:-

There are full text queries as arbitrary strings. It treats strings as words-bag for vector spacing model. It is processed by using standard vector-space retrieval methods.

Phrasal queries:-

It is the retrieved document with particular phrasing. It allows intervention stop words and stemming such as: "buy camera" matches "buy a camera" or "buying cameras" etc.

Proximity queries:-

There are list of words with particular maximum distance constraint between these terms. Example: "race" and "dogs" with 4 words per matching "…dogs can start race…". It performs stemming and doesn't count stop words.

Pattern matching:-

It is allowed as a query which matches the string than token word. It requires a sophisticated algorithm and data structure which inverts indices for retrieving efficient [6].

4. Data mining

It is understood as process of extracting useful, meaningful, unknown and ultimate comprehensive data from huge database. It is exhaustively and automatically explored larger datasets to uncover otherwise hidden relationships among data. Data mining technology is successfully applied in health, science, marketing and finance to aid newest strengthen and discoveries markets. In addition data mining methods are applied for discovering and organizing the information from Web. Fig.1.9 displays information pyramid that illustrates

data mining area. Data mining area includes knowledge, information and data as shown in the fig.1.9.

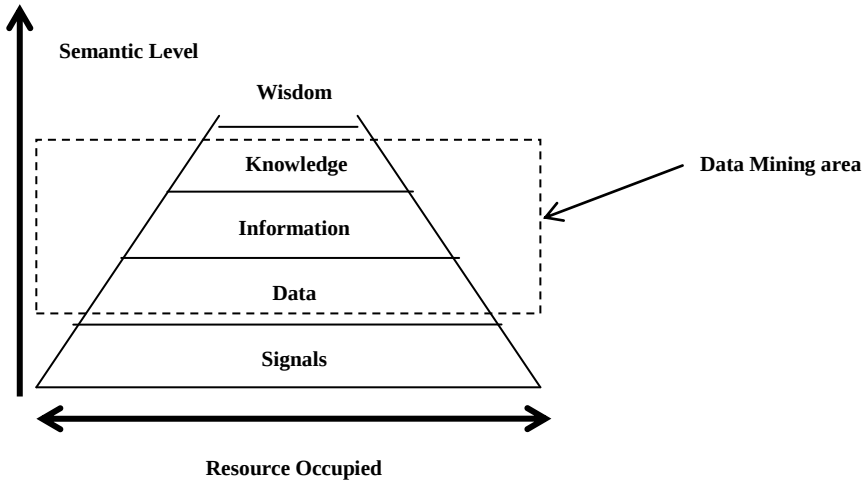


Fig.1.9: Information pyramid

It is understood as process of extracting useful, meaningful, unknown and ultimate comprehensive data from huge database.

It is analysis of large data quantities, exploration to discover meaningful, rules and patterns.

5. Machine learning

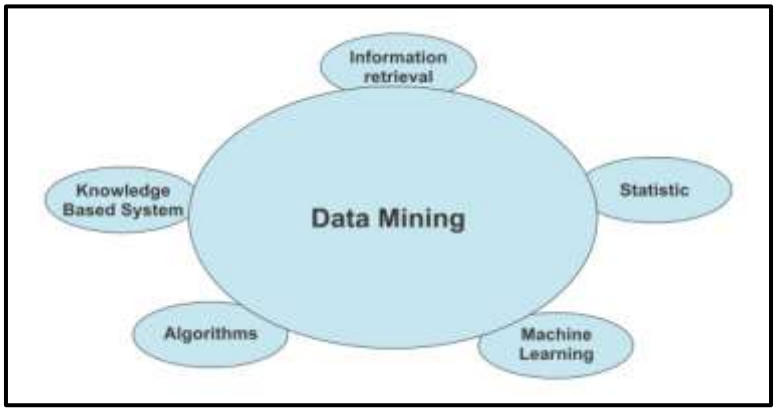


Fig.1.10: Machine learning

In the late 1980's machine learning has been applied in IR. The IR processes are divided in 4 different phases: indexing, feedback, query formulation, and comparison which

offers learning opportunities. Researchers are focusing on sub processes which try to increase the retrieval systems' performance. For example clustering technique is used to create the links between relational documents index and user modeling methods attempt to express the information possible from user queries. Diversity can cause problem when various collections have been searched for the complete query for one which is not compared directly with rank scores because every distribution and collection feature set is uniquely identified. The language model is employed to show the text which is paramount.

The machine learning and information retrieval group have developed the statistical natural language processing, core information retrieval and machine learning technologies, to help solve complex and challenging business problems [7].

But this literature mainly focuses on information retrieval in web crawling.

1.5 Web crawler

It is an automatic script or program which can download the contents from the World Wide Web (WWW) in an automatic manner. This process is known as web or spider crawling. Various legal sites in search engines use spider as a means for providing updated data. Other websites and web search engines can use web spider or crawler software for updating website contents or indexing of site's web content. Web crawler copies the pages which are visited for further processing by search engines for indexing the visited pages.

Fig.1.11: Web crawler [8]

1.6 Introduction to web crawling

Web crawler is a program that collects Hyper Text Markup Language (HTML) pages from the web by following links from the collected webpages. This process is known as web

crawling. Web crawling is the first and foremost stage in any web information retrieval system. Web crawlers follow the hyperlinks for web to get another websites. When user of search engine can enter query, search engine is indexing and returns relevant result based upon searching keywords. It is a process which automatically provides up-to-date and relevant data [8].

1.7 Working of a web crawler

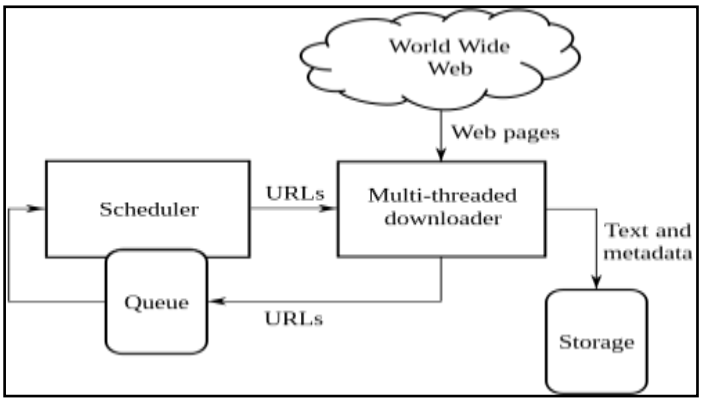


Fig.1.12: Working of a web crawler

A web crawler begins with an initial Uniform Resource Locators (URLs) list for visiting, known as seed URL list. When a crawler visit the URLs it identifies the page anchor texts (hyperlinks) and adds them into URLs list for visiting. This URL list is known as crawl frontier. URL in Frontier can be visited recursively with some policies. If crawler performs website archive to copy and save the info, the archive can store the data of view, read and navigate as on website live but is served as ‘snapshots'.

The huge volume can imply a crawler to download a limited set of web pages so there is requirement of prioritized downloads. The URLs crawled are generated by the server-side software making it difficult for web crawlers to avoid retrieving duplicate content [9].

1.8 Components of the web crawler

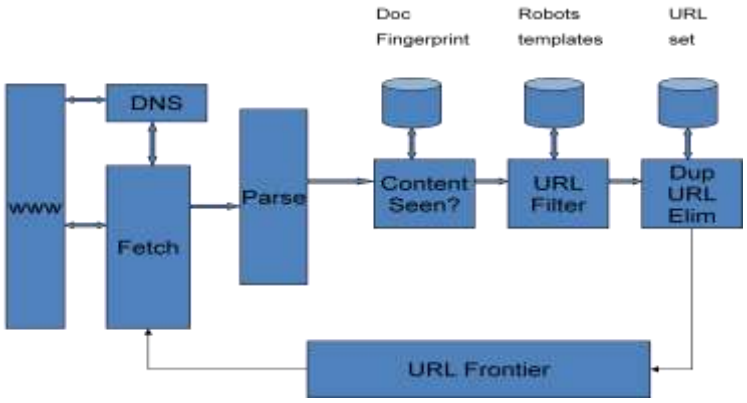


Fig.1.13: Components of the web crawler

Fig.1.13 displays web crawler architecture. The web crawler architecture consists of various components which are explained below:

a) Uniform Resource Locator Frontier (URL Frontier) contains URLs to be fetched in the present crawler.
b) A resolution module is a Domain Name System (DNS) which can determine web server for page specified by URL.
c) A fetching module which uses http protocol for retrieving at URL web page.
d) A parse module that can extract text or links set from fetched web page.
e) The duplicate elimination module which is determining the entry of the extracted link into URL frontier.
f) Crawling can perform anywhere from potential threads hundreds each of loops with logical cycle. The thread can run in process which is partition amongst several processes run individual nodes of distributed system. We start to assume it URL frontier is non-empty and placed.
g) A crawler thread can start taking URL from the frontier and fetch website page of URL using http protocol. The fetched page is written into temporary store where a number of operations have been performed on it. Next page is parsed and text link is extracted. This text is passed on indexer. Additionally, every extracted link go through a series of steps to determine if link is added into URL frontier.
h) The thread tests web pages having similar content already seen at other URL. The simple implementation will use fingerprint such as checksum.

i) URL filter can be used for determining if extracted URL can be excluded from frontier based on various tests.
j) The robots.txt file is for testing if the URL is to be considered for fetch from website, according to the robot restrictions.
k) URL is checked for further elimination: if URL already comes in frontier no need to check it further. When URL has been added into frontier which is assigned based on priority then it is eventually removed from the frontier for data fetch.
l) Some housekeeping task is performed by the dedicated thread. The thread is quiescent except which it wakes every few second to log the crawling progress deciding whether to terminate the crawl or checkpoint the crawl [10].

1.9 Web crawler features

A crawler must provide the various features:

a) Robustness: The web contains a server which creates a trap for spider by regenerating webpages which misleads the crawler into getting stuck on fetching multiple in a particular domain.
b) Politeness: Web servers contain explicit and implicit policies to regulate a crawler visit. These policies must be respected.

A crawler can be customized in the following ways:

a) Distributed: The crawling has the ability for being executed in distributed fashion across several machines.
b) Scalable: The architecture of crawler can permit to scale up the crawling rate to add extra bandwidth and machines.
c) Efficiency and Performance: The crawler system makes efficient usage of several system resources involving storage, processor and network bandwidth.
d) Quality: A fraction of website pages can be a slow utility to serve user query which then requires crawler to be biased towards fetching “useful” page first.
e) Freshness: In several applications crawling is operating in continuous mode, it is obtaining fresh copies of the last fetched pages [11].

1.10 Types of web crawler

1.10.1 Focused web crawler

It is a web crawling type, which tries to download the pages related to a particular topic. It makes a collection of documents that are relevant and specific to given topic. It is called as topical crawler because of way of working. It determines how a page can be relevant to specific topic and how to proceed further.

The focused web crawler contains major components such as: classifier can make relevant judgment on the crawled pages for deciding link expansion; distiller determines a centrality measurement of the crawling pages for setting priorities and crawling with dynamic reconfiguration controls [12].

Categorization of focused web crawler:

1. Keyword based approach

In [15, 16, 17, 18], keyword based approach is used to retrieve the information from the web. These papers produce extraction of Uniform Resource Locators (URLs) based on keyword or search criteria. They extract URLs from web pages which contain searched keyword in their content and consider only such pages as significant while ignoring the web pages irrelevant to the search.

a) Ontology based approach

Ontology based web crawlers use ontological concepts for improving their performance. So, it will be very easy to get relevant data as per the user requirements. Ontology is used for structuring and filtering the knowledge repository. In [15, 16, 17], ontology concept is used.

b) Link Semantic based approach

The link semantic is characterized by the semantics of the referred document. This system can retrieve the documents through speculation of the document relevance, based on the surrounding text of the link and the keywords in the link. In [16], link semantic based approach is used.

2. Exemplary documents based approach

In this approach, topics can be defined not using the keywords but using the exemplary documents. The exemplary documents are defined by Kimbrough & Blair as info documents which specify the intellectual structure of specific field of interest. These documents can be used as an alternative document representation technique for Information Retrieval (IR).

a) Data mining based approach

Data mining is a process for analyzing data from different perspectives and summarizing it into valuable information. The papers [19, 20, 21, 22, 23, 24] use various data mining concepts such as clustering, classification, fuzzy set, naive bayes *etc.* to retrieve only the subset of the web related to a specific topic.

b) Other miscellaneous approaches

The papers [25, 26, 27, 28] are based upon the various other approaches such as designing the focused web crawler using genetic algorithm, developing a focused crawler with web page change detection policy *etc*. The genetic algorithm manages the optimization of web crawling by choosing proper web pages to be obtained through the crawler.

1.10.2 Distributed web crawler

It is a distributed computing method in which search engines use several computers for indexing internet through web crawling and this work could be done by distributed web crawler. The central server can manage synchronization and communication as it is geographical distributed. It is basically used as page rank algorithm to increase quality and efficiency. The advantage of distributed web crawler is ingrained into robustness against system crashes or events are adapted to several crawl applications [13].

Types of distributed web crawler:

1. Map reduce based approach

Map reduce is a platform for processing huge data sets parallely on a cluster of employees. Map reduce concept is used in [29, 30].

2. Data mining based approach

Data mining is a process for analyzing data from different perspectives and summarizing it into valuable information. The paper [31] uses various data mining concepts such as clustering, classification, fuzzy set, naive bayes *etc.* for retrieving the most relevant content in a time efficient manner.

3. Scalability based approach

Scalability refers to an increase in the throughput of crawler as the number of crawler nodes increase. In [32, 33, 34], a lot of study has been done to identify challenges in improving scalability and thus suggest novel frameworks for the same.

4. Peer-to-Peer Network (P2P Network) and Distributed Hash Table (DHT) based approach

P2P network is a network of computer systems configured to allow sharing of some files and folders with everyone or selected users. In a decentralized distributed system, a special type of lookup service (or hash table) is called Distributed Hash Table (DHT), stores (key, value) pairs such that any node on the distributed system can extract value for any given key. In [35, 36, 37], distributed hash table and peer-to-peer network concepts are used.

5. Model based approach

The papers [38, 39] present model-based estimation of accuracy and availability of the distributed web crawler.

6. Other miscellaneous approaches

The papers [40, 41] are based upon the concept of developing vertical distributed web crawler using the crawling period strategy. There is a crawl manager which coordinates the resources and the number of tasks performed by each crawler in a distributed system. Each crawler is equipped to separately process websites having different update frequency.

1.10.3 Incremental web crawler

The process of revisiting and prioritizing Uniform Resource Locators (URLs) is referred as incremental crawling. On contrary, incremental crawler refreshes existing collection of the pages based on estimation of how pages can change. It exchanges low vital pages to newest and important pages. It resolves problem of freshness of pages. Fig.1.14 displays the incremental crawler architecture.

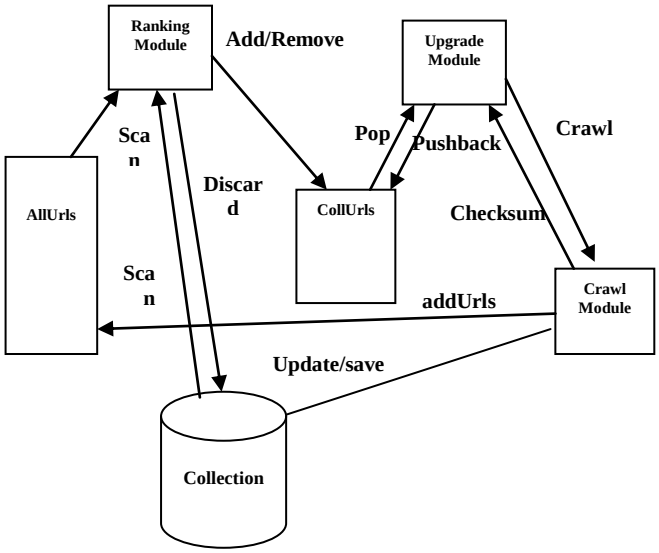


Fig.1.14: Architecture of the incremental crawler

Categorization of incremental web crawler:

1. Data mining based approach

Data mining is a process for analyzing data from different perspectives and summarizing it into valuable information. The papers [42, 43, 44, 45, 46] use various data mining concepts such as clustering, classification, fuzzy set, naive bayes *etc.* for retrieving most suitable content quickly.

2. Approach based upon improving the freshness using calculation of refresh time of pages

The papers [47, 48, 49] improve the freshness of the page content by calculating the refresh time of the web pages.

3. Other miscellaneous approaches

The paper [50] employs site level knowledge for incremental crawling of web forums and further designs a list-wise strategy by reconstructing the linking structure called sitemap for a specified forum site.

1.10.4 Hidden web crawler

The deep web is referred to World Wide Web (WWW) content which is not portion of web surface indexed by search engines. The hidden web is referred to web data collection which could be achieved by web crawler via interaction with web-based search form and not by traversing the hyperlinks.

Fig.1.15: Hidden web [14]

A large amount of data on web resides in database which is only retrieved by posting the exact queries. This data is called “hidden web” or “deep web”. Present day crawlers can crawl public indexing web *i.e* number of pages have been accessible by following hyperlinks but searching pages and forms that need prior or authorization registration have been ignored. In reality, it can ignore large quality of data which can be hidden behind search forms [14].

Types of hidden web crawler:

1. Tree based approach

For storing the downloaded web pages and for page analysis Dom Tree is used, which focuses mainly at differentiating and withdrawing plain texts and form features. This approach is used in [51, 52].

2. Domain specific based approach

Domain specific approach refers to crawling of a particular web domain. The papers [53, 54, 55, 56, 57, 58] uses domain specific approach.

3. Security based approach

Structured query language (SQL) injection is a code injection method used for attacking data-driven applications, in which important structured query language statements are added for execution. This approach is used in [59]. [59] represents a method for structured query language injection vulnerability detection based on deep web crawler.

4. Other miscellaneous approaches

The papers [60, 61, 62, 63, 64] based upon the concepts such as Open Grid Services Architecture - Deep Web Crawling (OGSA-DWC): middleware for deep web crawler using grid, Siphon++: deep web crawler for keyword based interfaces *etc*.

Chapter 2

Literature Survey

2.1 Strategies of information retrieval in web crawling

In web crawling, strategies of information retrieval are mainly divided into four main categories *viz*: focused, distributed, incremental and hidden web crawlers, which are further sub-divided into various sub-categories as shown in the fig.2.1. Fig.2.1 represents a hierarchical tree structure of strategies of information retrieval in web crawling.

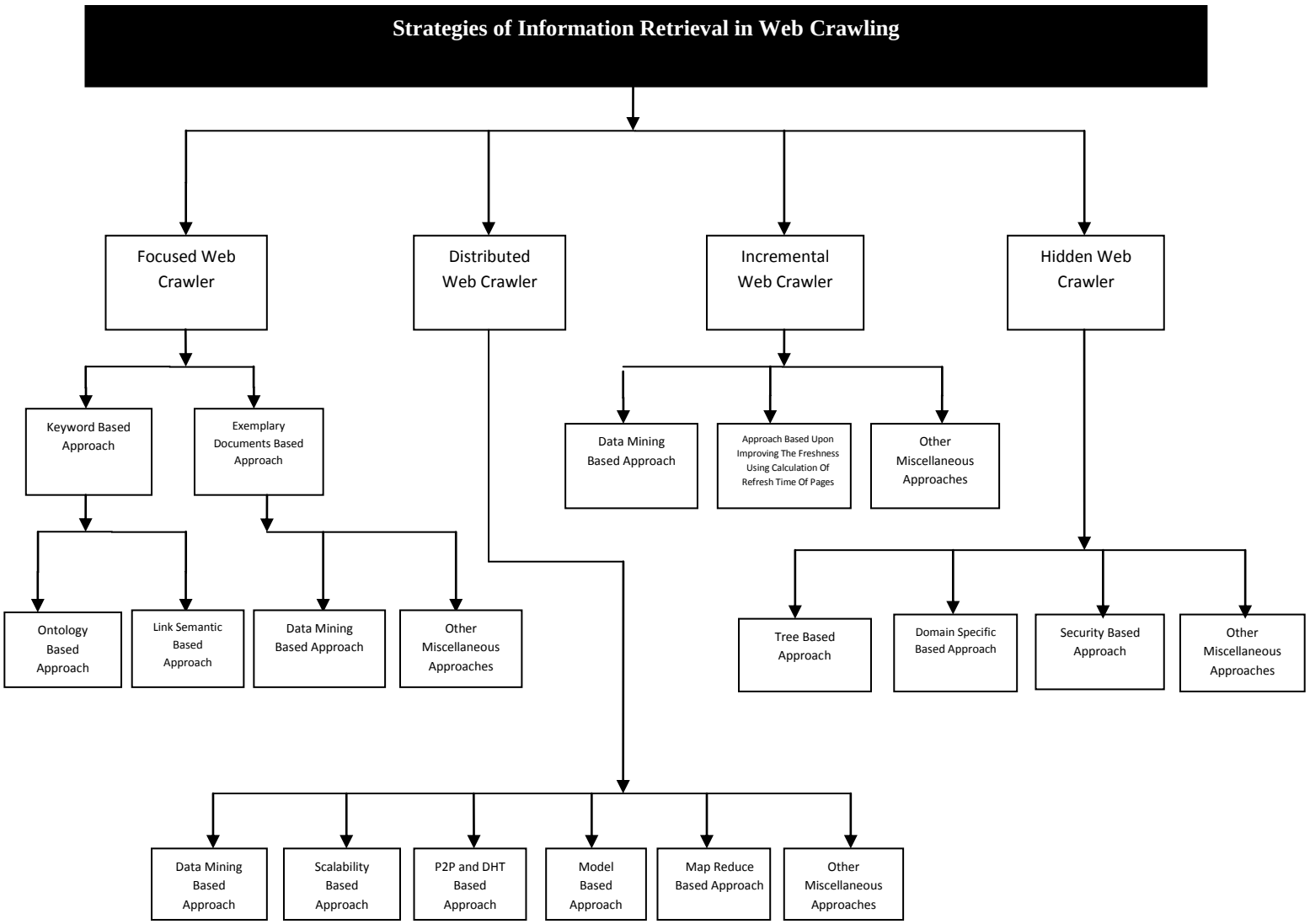


Fig.2.1: Strategies of information retreival in web crawling

2.1.1 Focused Web Crawler

Focused web crawler is used to collect the web pages, which satisfied some properties by prioritizing the crawler frontier and managing the hyperlink exploration process. Focused web crawler can be taken as a base crawler for various other representations such as keyword based approach, exemplary documents based approach, ontology based approach, link semantic based approach, data mining based approach and other miscellaneous approaches.

1. Keyword based approach

In [15, 16, 17, 18], keyword based approach is used to retrieve the information from the web. These papers produce extraction of Uniform Resource Locators (URLs) based on keyword or search criteria. They extract URLs from web pages which contain searched keyword in their content and consider only such pages as significant while ignoring the web pages irrelevant to the search.

a) Ontology based approach

Ontology based web crawlers use ontological concepts for improving their performance. So, it will be very easy to get relevant data as per the user requirements. Ontology is used for structuring and filtering the knowledge repository. In [15, 16, 17], ontology concept is used.

Gunjan and Snehlata [15] proposed an algorithm for ontology based internet crawler, which retrieved only relevant sites and made best estimation path for crawling that helped in improving the crawler performance. Employing only the keyword based focused web crawler, the proposed approach had dealt with information path and domain ontology to find out most related web content and pages according to user requirements. Ontology was used for filtering and structuring the repository information. The major benefit of keyword web crawlers over traditional web crawlers was that it did not take training for internal details or connectedness feedback as to how process was going on to act intelligently. There were two types of amendments which defined the results of each crawler: I) number of extracted documents was decreased. Analysis of the link and deletion of an appreciable number of irrelevant websites, II) the crawling process reduced the turnaround time. As a result of deletion of irrelevant websites, crawl load reduced.

Gunjan and Nikita [16] proposed a technique called keyword focused web crawler. Java tool was used in this paper. To improve the crawler's performance the concepts of domain engineering with relevancy decision mechanism, keyword-driven crawling and ontology were considered. This approach introduced the extraction of keywords based on Uniform Resource Locators (URLs) or criteria for best search. This scheme only downloaded the URLs whose web pages contained the searched keywords. Using some metrics such as link analysis algorithms (includes page ranking algorithm) and other URL prioritizing techniques to rank the URLs, this methodology was more optimal than the traditional web crawler in terms of search effectiveness. The sole focus of the approach was to have a selection policy (availing the Java tool) using which an internet user could save on his search time. The technique did not find the relevant web pages by using any other branches or art, as parent node was related to the "java"; the reason being the resultant data was much higher than the information path attained by word "art".

In 2015, Shubham *et. al* [17] proposed an idea for self-adaptive semantic focused (SASF) crawler. Major goal was to crawl the webpages related to their service. Two technologies:- ontology learning and semantic-based crawling were combined to have a crawl design which could help customers for accurate and efficient search of information from the internet. The approach strove to remove problems from existing system for SASF crawler with ontology learning which would help the user extract relevant information while using SASF crawler based search engine.

b) Link semantic based approach

The link semantic is characterized by the semantics of the referred document. This system can retrieve the documents through speculation of the document relevance, based on the surrounding text of the link and the keywords in the link. In [18], link semantic based approach is used.

Meiyappan *et. al* [18] discussed distinctive and novel focused crawler which was called as an Link Semantic Crawler (LS Crawler). Several algorithms and tools such as Simple Object Access Protocol (SOAP) based Web Application Program Interface (API), Porter Stemmer's Algorithm, Multithreaded Java Application were partook of this technique. The LS Crawler system retrieved information documents by speculating relevance of data

document based on surrounding text link and keywords in link. The motivation behind relevance in document was not only based upon probability technique but on semantics as well. LS Crawler extracted hypertext and measured the relationship with keyword search in particular domain to refer to ontology. This system had better recall due to utilization of keyword semantics in the link.

2. Exemplary documents based approach

In this approach, topics can be defined not using the keywords but using the exemplary documents. The exemplary documents are defined by Kimbrough & Blair as info documents which specify the intellectual structure of specific field of interest. These documents can be used as an alternative document representation technique for Information Retrieval (IR).

a) Data mining based approach

Data mining is a process for analyzing data from different perspectives and summarizing it into valuable information. The papers [19, 20, 21, 22, 23, 24] use various data mining concepts such as clustering, classification, fuzzy set, naive bayes *etc.* to retrieve only the subset of the web related to a specific topic.

In 2011, Duygu *et. al* [19] presented a technique based on segregation of links rather than a relevance measure to download individual web pages. Several algorithms and tools such as Link Scoring Algorithm, Simple Stemming Algorithm, C#, Naïve Bayes Classifier Algorithm, Java were used. Naïve bayes classifier was employed to classify Uniform Resource Locators (URLs) using a simple URL scoring optimization technique. The conclusion can be demonstrated which presented method performs better. Moreover, the simple stemming algorithm was developed for Turkish stop word list and Turkish language to improve the efficiency of the system. Five different techniques were applied and their performances were analyzed in terms of harvest rate (ranging from 0 to 1). A score higher than 0.8 qualified a page as relevant. Three distinct types of the naïve bayes classifiers were used for link scoring.

Qingyang and Wanli [20] proposed newest framework for focused web crawling based upon “relational subgroup search”. The relevance of unvisited pages in the crawl frontier was predicted based on some predicates followed by a subgroup discovery technique

for first-order classification rules. Relational rules with suitable confidence and support (calculated using an A* searching algorithm) were fed to a relational learner resulting in classification rules based on first-order logic. Use of relational knowledge representation and subgroup discovery renders novelty to this cralwer. Subgroup discovery technique aimed at finding subgroups with high proportion of the relevant documents. Preliminary evaluation experiments displayed potential of proposed focused crawler first-order. Several algorithms such as Best-First Algorithm and A* Search Algorithm had been used in this approach.

Qiang [21] proposed the OFC algorithm based upon fuzzy clustering and reinforcement learning theory to improve the harvest rate of a focused crawler. A combination of fuzzy centre averaged cluster method with naive bayes classifiers for calculating the fuzzy memberships was used for solving the valLie function map hyperlinks to next discounted rewards. A modification was made to the reinforcement learning method by Rennie et al and two online classifiers were proposed:- a label for web page and classification to anchor text in individual classes. Centre-averaged classification was used to classify the anchor text which further was classified by naive bayes theory. A novel fuzzy approach was designed as a combination of naive bayes method with center-averaged classification. Precision was employed for evaluating the crawl efficiency and harvest rate in the web mining.

Mali and Meshram [22] proposed a scalable architecture concentrating more on the page revisit policy and page selection policy in 2011. The content of the web page as well as the link structure was used to increase the page relevance and improve coverage of particular topic respectively. The crawler minimize the work of re-downloading the similar pages again. The technique captured the link structure of each webpage and worked differently according to the formatting of each page. The centralized database to download the Uniform Resource Locators (URLs) reduced the dependency of system. For this web crawler, to traverse web entirely and quickly was an unrealistic and expensive aim because of needed network and hardware resources. The main problem of focused crawler was to identify promise links to avoid off-topic searches and include only target documents.

Soumen *et. al* [23] proposed the newest hypertext resource search system known as focused crawler. The aim was to select the pages related to the given topics (defined using

exemplary documents). Two hypertext mining programs guided the crawler: the classifier evaluated relevance of hypertext content and distiller identified hypertext nodes (which can act as hubs to several relevant pages). The approach acquired, sought, maintained and indexed pages on the particular number of topics which represented the relevant narrow segment of web. The system proposed that the additional URLs were most probably in the smaller neighborhood of already found Uniform Resource Locators (URLs) which appeared to be same to examples. As an example, bicycle pages referred to the bicycle pages unexpectedly also to first-aid and red-cross pages. Several algorithms and tools were used: PageRank Algorithm, Berkeley Data Base (DB), Dual-Processor Pentium-II 333 MHz, Structured Query Language (SQL) and Random Access Memory (RAM) 256 MB.

In 2012, Mejdl *et. al* [24] proposed newest learning-based method for improved relevant prediction in focused web crawler. Naïve bayesian was chosen as the base prediction model, though any other individual prediction model could also be used. Nevertheless, the proposed technique was much accurate and efficient than related approaches. "Stock Market" was the sample topic used and extended learning-based relevant prediction model from two relevant attributes to four relevant attributes. The author implemented the method in prototype crawling and compared crawl performance with two crawlers. In this approach two more related attributes i.e. parent pages, surrounding texts, anchor text and Uniform Resource Locator (URL) words of visited URLs, were added. In this approach, level limit was fixed to 3, which signified that any path in the crawl tree that had three irrelevant consecutive pages have to be given up. This approach adopted naïve bayesian model but can be extended to sophisticated models. In this, represent the newest learning-based focused crawler method which used only four relevant attributes to take relevant of unvisited URLs. In this approach, several algorithms and tools that were used are given as: Porter's Stemming Algorithm, Naïve Bayesian Classifier Algorithm, Java and Breadth First Search Algorithm.

b) Other miscellaneous approaches

The papers [25, 26, 27, 28] are based upon the various other approaches such as designing the focused web crawler using genetic algorithm, developing a focused crawler with web page change detection policy *etc*. The genetic algorithm manage the optimization of web crawling by choosing proper web pages to be obtained through the crawler.

Anish and Priya [25] surveyed the focused crawler methods and classified them into five categories: structured base crawling, priority base crawler, context base crawling, leaning base crawling and focused crawler. This heuristic method was compared to native methods of web crawling focusing on comparison study between methods. The comparison was done in simulated web environment and result was based on the time taken to retrieve the relevant resultant documents. The main aim was achieved by prioritizing already crawled pages and by managing the exploration of hyperlinks. In this approach, web crawler architecture for exposing underneath secrets of the web crawling implementation was proposed.

Dvijesh *et al.* [26] proposed focused web crawler which acquired, indexed, sought and maintained the pages on topics which covered only a narrow segment of web. The approach verified several types of crawlers with cons and pros. A detailed study of the future directions to improve the efficiency of focus web crawler was done. The crawlers were divided into two parts: - preferential crawlers and universal crawlers. Further, preferential crawlers were divided into two parts: - the topical crawlers and focused web crawler. The topical crawlers also were of two types: - static crawlers and adaptive topical crawlers. The evolutionary crawlers, reinforcement learning crawlers *etc*. were included under adaptive topical crawlers and the best-first search, PageRank algorithms *etc*. were grouped under static crawlers. In this, several algorithms and tools were used: Hits Algorithm, PageRank algorithm and Java.

Hardik [27] in 2014 proposed a design for a personalized focused crawler with web page change detection policy, which narrowed down searching of new pages and therefore eliminated redundant and updated data. It proposed the method for multi-user focused web crawler which facilitated finding the changes in search page result since the most recent access. Here the system relevance of page was calculated according to the word frequency i.e. number of times search string was repeated in a web page. The web crawler could be customized to provide suggestions and results based on user profile, user described seeds or links visited by customer. In this approach, the change detection method was based on content of web pages.

Pranali and Prashant [28] proposed a focused web crawling using a genetic algorithm. In this genetic algorithm, the jaccard technique and data function were used to determine

related web pages. The major goal was to select the most promising links to maximize relevancy of newest unvisited Uniform Resource Locator (URL) by applying the genetic algorithm which tried generating optimal result and helped increase the accuracy and precision. With the use of genetic algorithm authors got the relevant links in less time. The main advantage of this approach was due to construction of collections of certain-area with high quality than fixed focused web crawling techniques and retrieval of relevant result in low time. Breadth First Search Algorithm was used in this approach.

2.1.2 Distributed Web Crawler

Distributed web crawler is a distributed computing software, in which by employing internet, search engines use several computers for indexing the content from World Wide Web (WWW) by web crawling. Distributed web crawler can be taken as a base crawler for various other sub-types *viz*: data mining based approach, scalability based approach, Peer-To-Peer Network (P2P Network) and Distributed Hash Table (DHT) based approach, model based approach, map reduce based approach and other miscellaneous approaches.

1. Map reduce based approach

Map reduce is a platform for processing huge data sets parallely on a cluster of employees. Map reduce concept is used in [29, 30].

In 2015, Do Le *et al.* [29] proposed UniCrawl as an effective geo-distributed crawler which minimized the inter-site communication costs. The limiting factor of distributed crawler architecture was higher infrastructure cost. For reducing their cost and high upfront investments, the authors proposed a geo-distributed crawler UniCrawl solution. UniCrawl arranged various geographical distributed sites. Each UniCrawl site design comprised a sequence of the map-reduce tasks executed on an industrial-grade key-value store. Also included was an ENSEMBLE novel layer that was federated available storage space at every site. When compared with the centralized architecture with crawling stretching over various places, UniCrawl displayed the improvement in performance of 93.6% in terms of bandwidth consumption of network and 1.75 in terms of speedup factor. The design was both scalable and practical, reducing the inter-site traffic by 93.6%. To lower cost the parse and fetch phases were merged in implementation of UniCrawl. UniCrawl was found to be 1.75 times faster than Nutch deployment. In this approach, several algorithms and tools employed were:

Java, OPIC Algorithm, Apache Gora, Nutch Version 2.5.3 and Ensemble. The contribution in the lines of code (LOC) was: 1.1 kLOC for Gora, 9.4 kLOC for Ensemble, 8 Core Xeon 2.5 Ghz Machine, 2.3 kLOC patch for Nutch, Ubuntu 14.04 64 bits, 8GB of Memory and 1Gbps Switched Network.

In 2014, Poonam and Rajashree [30] developed a distributed crawler for helping on-line shoppers for comparing their prices of products requested from individual vendors and take best price at one place. In this, the authors described the shopping agent implementation on the distributed web crawling while using map-reducing paradigm for crawling web pages. Moreover, it followed a method to build a web crawler which crawled only e-commerce websites with their content and products created separate database of content information from individual e-commerce websites. The projected crawler got all publicly accessible products of ecommerce websites from their site-map which saved all resource locators of products for Search Engine Optimization (SEO). The represented work used open source distributed framework called Hadoop for implementing web crawling. The proposed work implemented the shopping agent for accessing the sitemaps of e-commerce websites. The MapReduce paradigm was used for indexing part of data which made retrieval of search results exponentially faster. The PageRank algorithm was employed for ranking results. In this approach, several algorithms and tools used were: Page Rank Algorithm, Proprietary Algorithm, HTML and Breadth First Traversal Algorithm.

2. Data mining based approach

Data mining is a process for analyzing data from different perspectives and summarizing it into valuable information. The paper [31] uses various data mining concepts such as clustering, classification, fuzzy set, naive bayes *etc.* for retrieving the most relevant content in a time efficient manner.

Jose *et al.* [31] introduced evaluation of the scalable distributed crawler with geographical web partition. The main method was based upon the existence of several distributed crawlers to identify the most recent geographical zones. NetGeo was used for determining IP geographic locations. The evaluation displayed results for exchanging, downloading and relocating the time efficiency improvement, which was defined by using geographical location based crawl, when a set of crawlers needed to gather the pages in a pre-

established geographical scope of site-hash based method. The main problem identified two partition objectives to get all the optimizations: the reduced Internet Protocol (IP) web link graph to minimize the inter-partition links between IPs and geodesic distances graph to obtain the geographical focused topics to minimize proximity between IPs. The algorithm produced balanced web space. The tool Java Universal Network/Graph Framework was used in this approach.

3. Scalability based approach

Scalability refers to an increase in the throughput of crawler as the number of crawler nodes increase. In [32, 33, 34], a lot of study has been done to identify challenges in improving scalability and thus suggest novel frameworks for the same.

Milly *et al.* [32] proposed crawler called as LiDi Crawler. LiDi Crawler was a centralized distributed crawler application and comprised central node which managed individual crawling components. This approach represented distributed crawling topic that could be designed to avoid the approximations for limited network overhead and run relative inexpensive hardware. The proposed crawler aimed to avoid duplicate crawling data without effect on crawling efficiency. The implemented crawler for research project was focused on crawl effectiveness for minimizing the resource consumption. The conclusion was an effective, reasonable and scalable efficient crawler application. In this approach, recursion detection features in the LiDi Crawler feature did not exist in traditional crawlers. The approach displayed that the recursion detection outperformed techniques fixed-depth counterpart in 3 major aspects: effectiveness, redundancy reduction and retrieval coverage. In this approach it was displayed that it was possible to create efficient usage of network bandwidth when comparing to stand-alone crawler. The proposed crawler was scalable when several slaves were added for participation in crawl for retrieval of high data. Crawler's effectiveness was examined using precision.

In 2011, Sunil and Neelima [33] implemented DCrawler a distributed scalable web crawler. The main features of crawler were decentralization of tasks, platform independence, effective assignment function to partition domain to crawling and ability to cooperate with web servers with full advantages in fault tolerance and scalability. Some other features of DCrawler were: full distribution of every task, platform independence, scalability, tolerance

to failures. Simple tests with distributed web crawlers successfully displayed that the Dcrawler performed better than the traditional centralized crawlers. DCrawler introduced the newest ideas in parallel crawling and in domain of crawling – web server coordination. Various web portals could be downloaded and analyzed using DCrawler. The statistics for the same was extracted for multimedia files, Hyper Text Markup Language (HTML) pages, link structure and web sites. Finally an important component in DCrawler was a reliable failure detector which was used for timeouts to find crashed agents reliability. In this approach, several algorithms and tools used were: Java, Breadth-First Visit Algorithm and HTML.

Paolo Boldi *et al.* [34] proposed software to realize the efficient scalable and distributed web crawler. Consequently, the most important features recognized were tolerance and platinum-independence to transient failures. Additionally, their software made possible recovery from permanent non-destructive failures in simple manner. Instead of explicit implementation, the dedicated network protocol to inter-agent communication using remote technique invocation. In this several algorithms and tools used were: Java, Dijkstra Algorithm and Remote Method Invocation (RMI).

4. Peer-To-Peer Network (P2P Network) and Distributed Hash Table (DHT) based approach

P2P network is a network of computer systems configured to allow sharing of some files and folders with everyone or selected users. In a decentralized distributed system, a special type of lookup service (or hash table) is called Distributed Hash Table (DHT), stores (key, value) pairs such that any node on the distributed system can extract value for any given key. In [35, 36, 37], distributed hash table and peer-to-peer network concepts are used.

Fei *et al.* [35] proposed the designing of web crawler distribution on the grid platform. This crawler was based upon last work Igloo. This approach used the crawler of Igloo as single crawler to make IglooG. Informational services in system were for districting the Uniform Resource Locators (URLs) to balance loads of the crawlers. Informational services could be cyanided as peer-to-peer network. Igloo could be implemented with the JAVA which could be run in linux and windows *etc*. The authors explained the implementation of the distributed crawler based upon Igloo and simulated environment of grid for evaluating balance load on crawlers and determined the speed.

Fig.2.2 displays the architecture of IgooG.

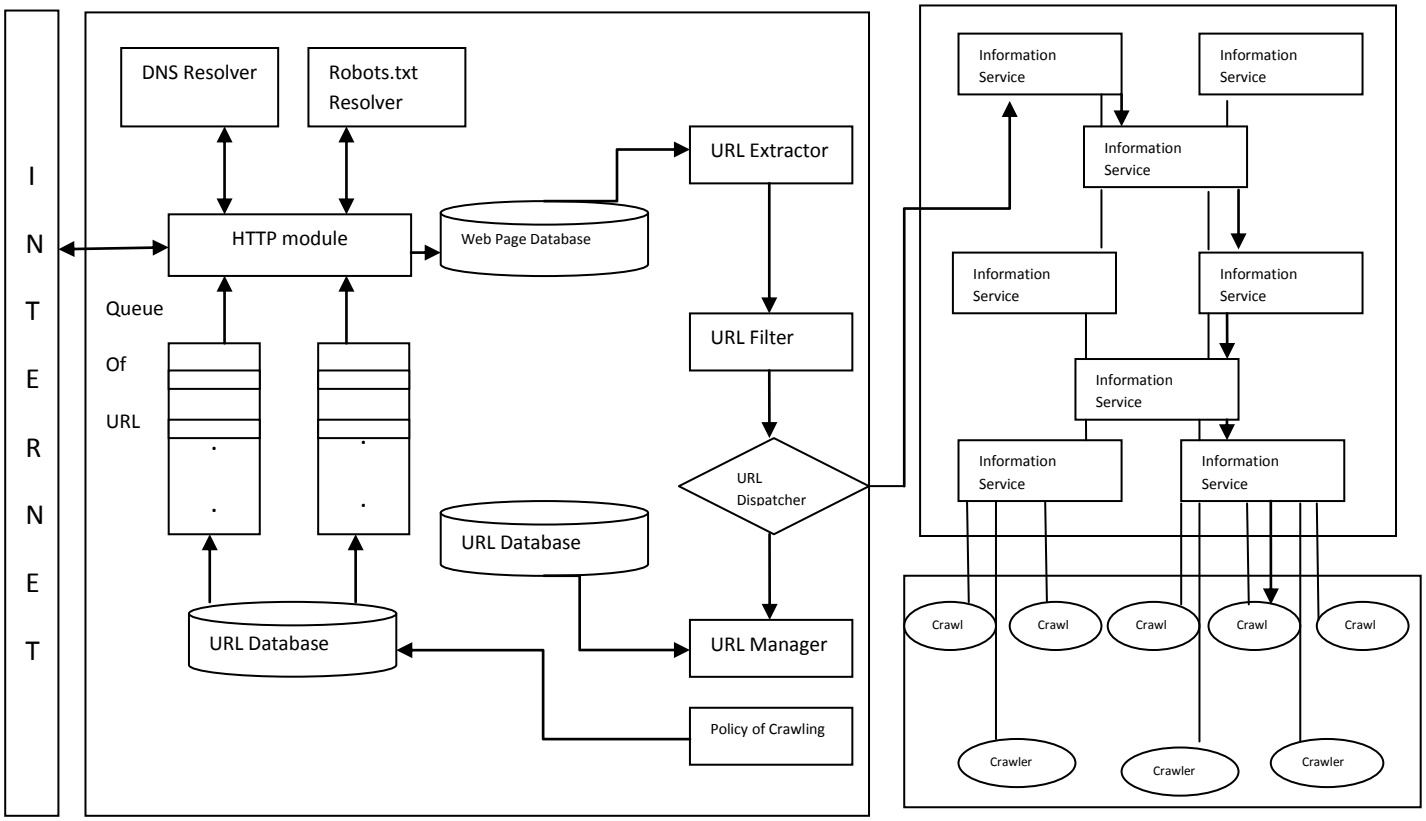


Fig.2.2: Architecture of IgooG

Both experimental results and theoretical analysis displayed the system was largely reliable. Grid service was adapted as a distributed environment service to improve quality of service. So, Igloo could scale up to whole web for fetching web documents tens millions. Large Scale Integration (LSI) controlled problems such as noise and synonyms by using statistical derived conceptual indices. The download speed of the simulator was 66 URLs and 660 KB/s. The average download speed of the IglooG scaled linearly as the number of participating crawlers increased. In this approach, several algorithms and tools used can be summarized as: Java, 100Mb/s Ethernet LAN and Seven 1.7GHz Pentium IV Machines moray.

Anup *et al.* [36] proposed peer-to-peer architecture, fully distributed for web crawling. The major aim of development system was to provide the alternative which was easily efficient, implementable, decentralized system and for indexing, querying, crawling and caching the web pages. The authors proposed an architecture which was easily

implemented on local network that followed peer-to-peer architecture and fully distributed. The major goal was to realize peer-to-peer web crawler framework and highlight features, credibility and advantages of system. The system called JADE followed full decentralized distributed architecture. The system comprised the peer-entities which formed atomic units of system and could be used as standalone or in network. The system did real-time index which meant pages passing crawler could be searched instantly. Kelondro database subsystem was employed to realize features and requirements. The use of censorship-free policy avoided all restrictions present in this approach, enabling coverage of World Wide Web (WWW) as well as of "deep web" or "hidden web". Also the use of the key based routing and distributed hash tables provided solid framework for distributed peer-to-peer network architecture.

In 2004, Boon Thau Loo *et al.* [37] proposed design and implementation of distributed web crawler. The authors engaged different crawling methods to investigate trade-offs in the crawl throughput, balance load on the crawlers, communication overheads along with crawl targets and ability to explore the network proximity. The authors represented the distributed crawler implementation while using relation query processor, PIER which was running over Bamboo Distributed Hash Table (DHT) and concluded by comparing the different crawling strategies on the Planet-Lab query live web sources. This approach focused on design issues of the distributed crawler that were important for proposed decentralized search applications. This approach represented the initial exploration of design space to build distributed web crawler over the DHTs. The crawler could be designed to run over DHT and by expressing crawl as query user-customizable refinements were permitted along with enabling the use of query processors like Telegraph as long as they supported recursion. Several tools used were: Peer-To-Peer Networks (P2P Networks), Distributed Hash Table (DHT) and Cumulative Distribution Function.

5. Model based approach

The Papers [38, 39] present model-based estimation of accuracy and availability of the distributed web crawler.

Seyed *et al.* [38] introduced Distributed Crawler for Rich Internet Applications (Dist-RIA Crawler) and explained the challenges and practical aspects faced to design it. Some technologies such as partial Data Object Model (DOM) and Asynchronous JavaScript and

XML (AJAX) updates made problem of crawling and time consuming. For reducing time to crawl RIA represented the newest distributed algorithm for crawling in the parallel with several systems known as Dist-RIA Crawler. The Dist-RIA Crawler could also use JavaScript event in DOM structuring for partitioning of search space. The distributed crawler defined as Dist-RIA Crawler achieved a parallelism such for every vertex; every crawler has explored particular subset of edges. Dist-RIA Crawler had deterministic finite RIAs as targets. This approach was based upon Breadth-First-Search (BFS) and was optimized for avoiding repetition execution of the similar AJAX calls. But this approach was not elastic to the available resources. Several algorithms and tools such as: JavaScript, Breadth-First-Search (BFS) Algorithm, PHP 5.2.10, Hyper Text Markup Language (HTML), Apache Web-Server, MySQL 5.0.77, Intel Xeon CPU E7330 @ 2.40GHz, Linux Kernel 2.6 Operating System, C#.NET use .NET 4 framework, 4GB of RAM, V8-Engine, Windows 7 Enterprise Operating System, 1GB of RAM, Intel Core 2 Due and Hyper Text Transfer Protocol (HTTP) for TCP channels use 10G-bps local area network were used in this approach.

Mitra *et al.* [39] proposed an accurate distributed and model-based web crawling architecture that is based upon the UbiCrawler. Stochastic networks were be used to model the crawling. In this, the dependability or accuracy measures were discussed along with classification of the failure and faults states of crawling and representation of the crawler stochastic model. The authors restricted scope of availability and extended the crawler model in measurement support. The authors had data in paper as input to models and used a partition policy in model. The authors included platform faults and designed proposed distributing web crawler in the stochastic activity networks using Mobius tool. In this approach, several tools were used which were given as: Stochastic Activity Networks, Hyper Text Markup Language (HTML) and Mobius.

6. Other miscellaneous approaches

The papers [40, 41] are based upon the concept of developing vertical distributed web crawler using the crawling period strategy. There is a crawl manager which coordinates the resources and the number of tasks performed by each crawler in a distributed system. Each crawler is equipped to separately process websites having different update frequency.

Vladislav and Torsten [40] proposed implementation and design of higher performance distributed web crawler which ran on the network of work stations. The crawler could scale to various pages. This approach described the efficient techniques for achieving the performance battle and a design for the software system architecture and higher system performance. This scheme described the implementation and design of optimization system on workstations network. This approach focus was on wide breadth crawler written in c++ & python which gave the flexibility and alleviated number of problems. In this approach, few fairly powerful downloaders which opened hundreds of the asynchronous connections while threading for queue opened one connection using synchronous I/O. Atrax scaled to replicate various verticals slices while system layers having various horizontal slices such as Uniform Resource Locator (URL) queuing, application in manager *etc*. In this approach, parsing for index terms could be integrated with download speed and crawl system could be limited by index speed. In this approach, authors focus was on network efficiency and I/O aspects such as computer and crawling speed scalability issues and no. of participated nodes. Several algorithms and tools such as: Breadth-First Crawl algorithm, Berkeley DB5, C++, Merge Algorithm and Python were used in this approach.

In 2010, Bing *et al.* [41] introduced the distributed crawler with the use of crawl period based strategy distribution. In this approach, distributed template customized vertical crawler that could be special used for crawling internet forums was described. In this approach, the crawl period strategy distribution and crawling manager could be coordinated with crawling quantity tasks. This crawler was designed for forums for news websites and blogs. This crawler could be high procession forum crawler based upon vertical crawling. The distributed vertical crawler architecture could be client server. This approach used the hash method but in dynamic path so crawling avoided downloading unchanged and repetitive Uniform Resource Locator (URL). Experimental results indicated that the parallel working of crawling in the distributed system could increase crawl performance. The performance of total distributed system could be improved greatly. In this approach, several algorithms and tools used which were given as: Hyper Text Markup Language (HTML), Hash Algorithm, and JavaScript.

2.1.3 Incremental Web Crawler

The process of revisiting and prioritizing Uniform Resource Locators (URLs) is referred as incremental crawling. Incremental web crawler can be taken as a base crawler for various other representations *viz*: data mining based approach, approach based upon improving the freshness using calculation of refresh time of pages and other miscellaneous approaches.

1. Data mining based approach

Data mining is a process for analyzing data from different perspectives and summarizing it into valuable information. The papers [42, 43, 44, 45, 46], use various data mining concepts such as clustering, classification, fuzzy set, naive bayes *etc.* for retrieving most suitable content quickly.

In 2014, Pallavi and Rajiv [42] proposed crawling mechanism that could be generated to support the processing systems and data mining and also get the history of web's content. This approach explained a different crawler which executed easy processing with focus on collected information of ignored images, style sheets, web pages, java script and document files. This crawler could use the database system and directly saved the data into the system for maintenance and exploration of the acquired data from crawl. This crawl could be capable of adapting their frequency to changing rate of webpage content. This approach provided the implementation of robots exclusion protocol that let the web masters declare part of sites off limit to web crawlers. The low bound of web accessing intervals of crawler was 10 seconds.

In 2003, Hadrien *et al.* [43] introduced a perspective of creating an efficient and effective way of increment crawler. The main purpose was the optimization of the incremental crawler using the actions of user on resulting page which was returned by the search engine. Moreover, the authors introduced the newest perspective for improvement in revisit process of web crawler. Along with this was presented a method where a user can choose the pages of web crawler that were updated. These pages were not famous or not updated frequently by crawler. A poisson processing was used for modeling the changes of webpage.

In 2001, Jenny *et al.* [44] proposed a design of web crawling implementation for Almaden's IBM WebFountain project and described the optimizing model to control the crawling strategy. The WebFountain crawling outlined the design as keeping local copy of upto 1MB of text every page, plus metadata in the repository for mining and indexing *etc*. In this, NEOS were used as public server system that ran the experiments with various NLP solvers on the models and searched the standardized NLP package and MINOS to give the most and best reliable results. In this, contrast the model would be adaptive, flexible which was totally based upon entire web and serves graceful for their growth. The features of web crawling were too distinguished from previous crawlers which were fully incremental and distributed. C++ tool was used in this approach.

Qingzhao and Prasenjit [45] identified web pages features which would correlate to changing frequency. The authors design used crawl algorithm for collected the web pages clusters which were correlated to changing frequencies and this was examined in the history. The authors ran the experiment on the dynamic web content set which was about 300,000 different distributed Uniform Resource Locators (URLs) over 210 web sites. The experiments displayed that the web pages in cluster change at same frequencies. The authors proposed the web crawl policy model which measured the bottom line improvement known as user quality search results. The results indicated that the Cluster Level Sampling (CLS) method was more efficient and effective than crawl methods for maintaining the final quality ranking and satisfying user's queries of search engine. In this approach various algorithms and tools used which were given as: Repeated Bisection Clustering Algorithm, PageRank Algorithm and Cluster Level Sampling Algorithm.

Christos Bouras *et al.* [46] described the crawling method which was created to support processing systems and data mining to attain the history of web's content. The design and structure of mechanism was easy and simple but experimental results were displayed simply which made the crawler strong with fixed mechanism. This approach described distinct crawler which was executed as a simple and easy process focusing on content collection of ignore images, style sheet, java script, web pages and document files. The crawler could describe friendly measures to pages it visited. The second vital characteristic of the web crawl system was adaptability. The crawler described in this approach used the time limit to avoid their traps.

2. Approach based upon improving the freshness using calculation of refresh time of pages

The papers [47, 48, 49] improve the freshness of the page content by calculating the refresh time of the web pages.

Niraj *et al.* [47] introduced an alternate way to manage the process of visiting of a web site again. In this approach to synchronize or balance the frequency of revisiting a website, a novel architecture and a moral mechanism for incremental crawler was proposed. Also the issues such as: implore quality of the local collection and keep the freshness of local collection was eliminated in this approach. Overall architecture of this approach was incremental and scalable as well which could be correlated at crawl workers level and Uniform Resource Locator (URL) queues that were responsible to download documents from the website. This newly purposed architecture of incremental web crawler was responsible for maintaining the freshness of document at search engine sites and also for managing the process of visiting a web site again .This proposed architecture was highly suitable for parallel . This approach employed ecology of crawl workers to the web site which was responsible for managing the revisiting of a web site.

In 2008, Ashutosh *et al.* [48] presented a good approach to build an efficient incremental web crawler. This approach updated local collection of web and/or its database selectively instead of refresh collection in the batch mode periodically which led to improvement in "Freshness" of document and brought newest webpages in an efficient timely manner. This approach dynamically calculated the refresh time of the page for the next update and also detected web pages which needed updation frequently. This approach adjusted the website visit frequency by assigning a priority to a site dynamically. A structure for calculating the dynamic priority for any website was developed. This scheme purposed the concept of dynamic refresh time. A novel approach for managing the volatile information was introduced to increase the efficiency of the crawler and to reduce the traffic on the web. An alternative approach to optimize the frequency of visits to sites was proposed.

Fig.2.3 displays the system architecture of self-adjusting refresh time based incremental crawler.

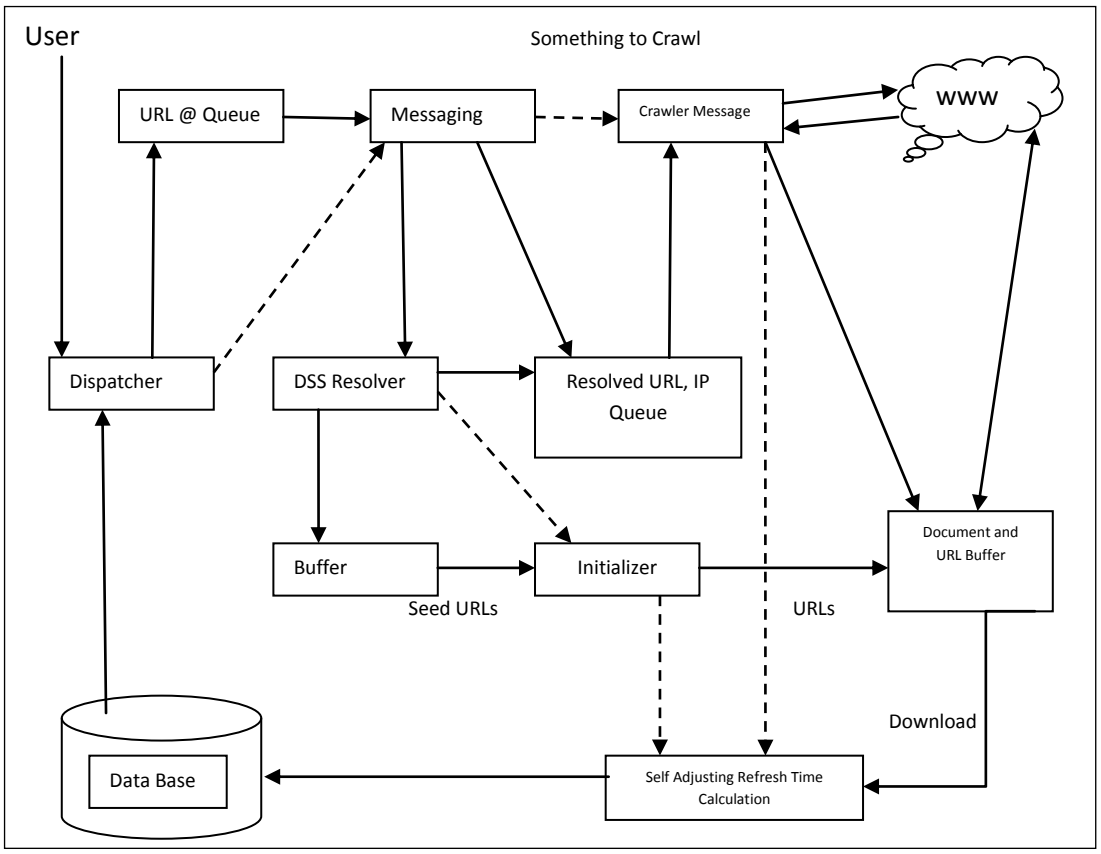


Fig.2.3: Architecture of self adjusting refresh time based incremental crawler

This modified version of crawler optimized the refresh rate frequency for an incremental crawler which helped in improve the quality of collection without obtaining much network traffic. By providing only useful data to the user, data environment could be achieved and network traffic could be reduced.

Hector Garcia-Molina and Junghoo [49] proposed a way to build an effective incremental crawler. They proposed an architecture for the incremental crawler by combining the best designs. This approach described how to create an effective incremental crawler by studying how web pages evolved over time by conducting experiment on the 720,000 webpages for months. The authors used their operational web base crawler for conducting experiment. The authors identified several design choices for constructing an incremental

crawler and analyzed how these choices affected the effectiveness of a crawler using their experimental data. The authors proposed architecture for an incremental crawler based on their observations which customized frequency of webpages depending upon how frequently authors changed and maintained only important pages. This approach derived new version of web crawler. The methods used for this experiment would drop a light on how this incremental crawler should work and which stats-gathering methods it should consider. For this experiment the authors adopted an active approach for crawling within a page window. In this experiment the researchers crawled 3,000 pages at each site. By adopting researchers approach, it is believe that crawler quality and freshness of its collection was improved significantly. In this approach several algorithms and tool used which were given as: PageRank Algorithm and Java.

3. Other miscellaneous approaches

The paper [50] employs site level knowledge for incremental crawling of web forums and further designs a list-wise strategy by reconstructing the linking structure called sitemap for a specified forum site.

Jiang-Ming *et al.* [50] proposed about the drawback of increment crawl of website forums that was basic but a challenging step in various website applications. Rather than treating each page independent of one another they proposed strategy of list-wise by using knowledge the site-level strategy. Such kind of information was used for constructing the link structure again which was called as sitemap for web forum site. These experiments resulted in bandwidth utilization, timeliness of their crawler and also showed favorable performance of coverage on 18 various forums. The proposed solutions of these experiments were evaluated on the mirrored data set which contained 5,407,854 individual posts and 990,476 pages that were published from the March 1999 to June 2008. This approach was designed from two parts one was online crawler and other one was online mining. The site-level knowledge was thus incorporated in the process of online crawling. To make a distinction between the advantage of bandwidth control and benefit of list-wise strategy authors implemented two variations of the proposed method one was list-wise strategy + bandwidth control and other was like-wise strategy. In this approach, authors used many different posts for measuring the crucial data in forum sites. Besides list-wise strategy + bandwidth control performed considerably better than other approaches of timeliness. List-wise strategy +

bandwidth had best policy if compared to list-wise strategy which additionally improved the performance. List-wise strategy + bandwidth control improved performance for indexing-of thread pages when compared to list-wise strategy. In this approach several algorithms were used which are as follows: Single Linkage Algorithm, PageRank Algorithm, Depth-First Search Algorithm and Breadth-First Search Algorithm.

2.1.4 Hidden Web Crawler

The deep web or hidden web refers to the web data which can be obtained by the web crawler via an interaction with web-based search form and not by following the hyperlinks. Hidden web crawler can be taken as a base crawler for various other types such as: tree based approach, domain specific based approach, security based approach and other miscellaneous approaches.

1. Tree based approach

For storing the downloaded web pages and for page analysis Dom tree is used, which focuses mainly at differentiating and withdrawing plain texts and also form features. This approach is used in [51, 52].

In 2011, Weicheng *et al.* [51] presented newest architecture of the web-crawler that aimed to analyze and get the information from web pages by using interfaces form. This crawler made some inventive parts such as extending their function into Asynchronous JavaScript and XML (AJAX) form to deal with the approach or differentiate distinct sites with similar Uniform Resource Locator (URL) while using improved Bayesian classification and mainframe extracting module. Dom tree could be used to create visual analysis and work of the pages which were already downloaded. The main aim was to expect the work for reaching contained various points. This approach considered mainframe of webpage for search form which made work flow difficult and precise if comparing with others that would take number of useless parts into consideration. To store download web pages Dom tree was used which was beneficial for extracting and searching. The authors used an advance ranking key-word strategy which selected generally Dom tree concerned with two points which were named as fitness for direct contribution of operation benefiting from their tree structure and second one convenience for discern passages and forms by tags indexing. In this approach, authors covered an inventive architecture of web crawler that was able to extract data from

both deep website and surface website. In this approach, several algorithms and tools were used which were given as: Naive Bayesian Classifier Algorithm, Bayesian Decision Model, Asynchronous JavaScript and XML (AJAX) and Hyper Text Markup Language (HTML).

Anita *et al.* [52] proposed a new vision based approach of Deep iCrawl to extract information from deep web. Deep web iCrawl was divided into 2 phases. In one phase, this approach included a query translation and analysis and in the second phase, covered vision-based data extraction from deep website pages which could be created dynamically. There were various methods to extract the deep webpages but this proposed approach aimed to overcome inherited limits of former. This approach compared data and items in particular ordering. In this approach, there were some visual similarities of data items which were exploited for making extraction into data records generic or effective. A web page might be containing audio, image, video, text *etc.* but the approach mainly focused on fonts and web page layout. This approach used Vision-Based Page Segmentation (VIPS) algorithm for conversion of deep webpage in visual blocking tree. This approach efficiently extracted data items and the deep website data recording using visual features. In this approach, authors constructed the deep web pages database of distinct domains that were updated commonly. Several algorithm and tools were used such as: Extensible Markup Language (XML), Visual Basic Scripts (VB Scripts), Vision-Based Page Segmentation (VIPS) Algorithm, Data Alignment Algorithm and Java Scripts.

2. Domain specific based approach

Domain specific approach refers to crawling of a particular web domain. Domain specific approach is used in [53, 54, 55, 56, 57, 58].

Manvi *et al.* [53] presented extraction of hidden web data for specific domain. This approach focused on the extracting hidden data behind html search forms. This approach used semantic mapping to fill the html search form by using domain specific database. The crawler here was guided by a set of data values based on domain of interest and crawled web focusing on pages relevant to a given domain. This approach mainly focused on the query submission. This approach entitled “domain based semantic hidden web crawler” provided a simple and efficient technique to extract data from hidden web database of interested domains. This approach was implemented for book domain taking websites having single

attribute search forms and multi attribute search forms. From the results of observation there was a performance improvement in percentage of forms filled or number of valid webpages retrieved for comparing traditional Hidden Web (HW) crawlers. This approach could not extract data from websites having unstructured databases. This approach lacked in semantic mapping of query while extracting data from hidden web databases.

In 2010, Rosy *et al.* [54] proposed architecture which introduced method for continuous refresh/ update of website hidden repository. In this approach, framework represented the repository updates of search engine to re-crawl webpages which could be frequently updated. This approach could be used for adjusting the mechanism time period between the two successful revisits of web crawler based upon the webpage. The architecture of hidden increment web crawling was a representation for continuously updated/refreshed hidden web repository. In this approach, design of hidden increment web crawler discussed that not only crawling hidden web but maintaining repository updated with new/updated pages was important. The represented hidden increment web crawling implemented the usage of .Net technology on the windows platform and took snapshots for hidden increment web crawler. Various experiments were conducted on airline and books domains and starting results were promising. The experiments conducted over real web sites indicated that the crawler always kept the repository fresh with large freshness rate. The projected hidden increment web crawler was updated for hidden webpages which had been crawled already by hideden web crawler to maintain repository freshness. In this approach, several tools used which were given as: .Net Technology, Windows Platform.

Nupur and Shalini [55] introduced novel approach to extract web query interfaces by using query condition rules. This approach deduced the schema of every query form from query condition rules. This approach proposed a novel approach of extract query forms considering Hyper Text Markup Language (HTML) forms and extracting query forms by HTML tag used for forms. This crawler enabled methods such as extracting hidden website content and indexed the analysis. The mined data was used for categorizing or classifying hidden databases. The major problem of crawler was that it needed high amount of memory space and a large network for maintaining fraction of whole internet. The solution of drawback was multi-attribute query interface. This method improved quality of query form extraction.

In 2011, El-Desouky *et al.* [56] implemented a newest algorithm to extract the labels from multi-attribute searching form fields. Embedding of projected algorithm into Hidden Web (HW) framework crawling was evaluated via experiment on real websites. The main conclusion could be demonstrated by precision and accuracy of presented method for sites considered. Main objective of newest label extraction algorithm was to correctly extract the labels from Hidden Multi-Attribute (M-A) web forms. This scheme for automatic generation query to Single-Attribute (S-A) forms was outlined. The authors introduced latest automate label extraction algorithm which correctly extracted labels from Hidden Multi-Attribute (M-A) web forms and determined the labels which had adjacent form fields of major algorithm contribution capability for distinguishing the forms.

Dilip kumar *et al.* [57] presented a technique for domain hidden web crawler. The proposed technique was based upon domain specific crawling of World Wide Web (WWW). Experiential results verified that the proposed approach was quite effective in crawling the hidden web data contents. This approach concentrated on the searchable interfaces for extracting dynamic hidden web contents behind the databases. These databases could be accessed via html searchable forms. There were various challenges associated with the discovery of searchable interfaces such as to find the algorithm to define the domain of the databases and how to fill the forms automatically to extract out the most useful information for the users. To complete these challenges domain specific hidden web crawler was proposed in this approach which did not require human intervention to crawl the hidden web information. The proposed crawling technique for domain specific hidden web crawling contained knowledge base which facilitated the extraction of useful and relevant information to the user through automatic form filling.

In 2010, Komal *et al.* [58] introduced Domain Hidden Web Crawler. The effectiveness of projected framework was evaluated via experiments with the use of dynamic websites. This technique could be used in domain hidden web crawler which automatically downloaded the searching interfaces and fill automatically. After filling the submission form it got pages as response. The projected hidden web crawler aimed to automatically process the view, fill, search and submitting of search forms and also analyze pages response.

Extensive experiments were conducted from several source domains on web with aim for evaluation the accuracy of projected specific-domain hidden web crawler.

3. Security based approach

Structured query language (SQL) injection is a code injection method used for attacking data-driven applications, in which important structured query language statements are added for execution. This approach is used in [59]. [59] represents a method for structured query language injection vulnerability detection based upon deep web crawler.

Xin *et al.* [59] proposed a technique of Structured Query Language (SQL) injection vulnerability based detection on the hidden web crawling and implementing detection system with reason of raised website page coverage and enhanced SQL vulnerability injection detection ability of website scanner. The authors combined the authorization with crawling model and search structured query language vulnerability to simulate website attack and analyze the response information. In structured query language injection vulnerability system, 30 attacking codes were included. Moreover, this approach represented deep web crawler technology based upon authorization access data table. To understand the affect of proposed methods structured query language injection detect system in c# was used. The authors attributed the results capability into hidden web crawler of system which was implemented. In this approach, several tools were used such as: Asp, C#, Structured Query Language (SQL) Server 2005.

4. Other miscellaneous approaches

The papers [60, 61, 62, 63, 64] are based upon the concepts such as Open Grid Services Architecture - Deep Web Crawling (OGSA-DWC): middleware for deep web crawler using grid, Siphon++: deep web crawler for keyword based interfaces *etc*.

In 2014, Sonali and Komal [60] presented several deep web crawlers developed for giving advantages and techniques to overcome difficulty included in integration method for selecting the surface approach as success and discuss hereafter. The authors defined the several web crawlers which could be developed to content surface in hidden web. The web crawler could differentiate based on underlay methods and behavior on different domains and

search forms. The experiment result was displayed as an adaptive technique with remarkable performance in various cases. It was not scalable into hidden web databases in different domains. It did not consider the forms detect inside resulting pages. Query was selected only with use of hierarchical category such as in Yahoo! which would not consider the flat classification. It only dealt with full text searching forms.

Satinder [61] presented the information about hidden web and several faced challenges while designing the hidden web crawlers. This approach discussed the challenges faced by the search engine designers while crawling the contents of hidden web. There were two fundamental characteristics of hidden web. The first one was the scaling of the web i.e. the hidden web is 400 into 550 times much larger than the Publicly Indexable Web (PIW). The hidden web was estimated to have 550 billion web pages equivalent to 7500 terabytes and was growing day by day. The second one was the need for crawlers to handle search interfaces that were designed primarily for humans. The traditional crawling did not distinguishing between pages with and without the forms.

Karane *et al.* [62] proposed the crawler for automatic retrieval of website content hidden behind keyword- interfaces. According to leverage goal with high-quality data in large unexplored part of website authors projected newest method for automatic hidden data retrieval. The Siphon++ performance was heavily dependent upon quality of sample. Siphon++ obtained the large coverage value for earlier iterations. Siphon++ attained the comparable performance to baseline. The best performance was obtained from Siphon++ from the first queries which was due to fact that Siphon++ could be built with the sample of indexing before staring the crawl whereas baseline could build sample as pages were be crawled.

Smita and Kriti [63] presented the methods and tools of crawling the web that was hidden beneath the surface. Deep web had plentiful information contained in it. This approach was a repository of very useful contents that were important for researchers at many levels. To use these resources there was need of an effective technique to take the desired and relevant content that was lying beneath the surface website i.e. deep web. Although some very useful algorithms and softwares were designed to explore the hidden web yet there was

much scope of finding new methods of crawling called deep web that can be cost and time effective.

In 2008, Jihwan *et al.* [64] proposed Open Grid Services Architecture - Deep Web Crawling (OGSA-DWC): middleware for deep web crawler using grid. It supported middleware functions fundamental for simple use and idle compute resources in world and crawling deep web. The authors proposed deep web crawl method using middleware. The middleware could be helpful for developers who implemented grid-based deep web crawling systems.

Chapter 3

Problem Statement

After reviewing the literature of information retrieval and its various categories for web crawling, it has been analyzed that the literature can be categorized under four major groups *viz:* focused, distributed, incremental and hidden web crawlers, which can further be sub-divided into various relevant sub-groups. As per the exhaustive literature survey conducted, there exists a scarcity of the work in which the said literature can be divided on the basis of a large set of 21 parameters and also that comment on the usage of particular type of web crawler in specific circumstances. This extensive study led to the experimental implementation of the basic focused and the distributed web crawlers based on which 4 parameters to distinguish these crawlers were identified.

Chapter 4
Methodology

The proposed work addresses the comparison between the conventional and the modernistic approaches *i.e* the focused web crawler and the distributed web crawler on the basis of four parameters *viz*: number of URLs crawled by focused and distributed web crawler, number of iterations taken by both crawlers, time taken by both crawlers for crawling the sites, number of hosts and type of threading used by focused and distributed web crawler.

Setting up the environment for focused and distributed web crawler:

1. Install the following software on the system, which are required for running both the crawlers.

1.1. Apache Nutch 1.10

1.2. Apache Hadoop-Core 0.20.2

1.3. Apache Solr 4.10.2

1.4. Java 1.8.0_31

1.5 Eclipse Luna IDE

1.6 Apache Ant 1.9.2

1.7 IvyDe 2.2.0

1.8 Subclipse 1.10.9

1.9 Maven m2e 1.5.2

1.10ygwin64

2. Implement the focused and distributed web crawler.

3. Select the Seed Uniform Resource Locators (Seed URLs) *i.e.* http://www.thapar.edu, https://www.mongodb.org, https://wiki.apache.org/nutch/ and http://agrifarming.in/. Input these URLs one by one in the crawler and crawl that URL.

4. Now, for each seed URL, record the number of hosts, number of URLs, the number of iterations, number of threads (or number of queues) and the total time taken for crawling the websites; for both crawlers.

Chapter 5

Experimental Results

This chapter focuses on the tools which are used for implementing the two strategies of information retrieval in web crawling *i.e.* the focused and the distributed web crawler. Moreover, the working of these two well known strategies has also been implemented.

5.1 Tools which are used for the implementation of focused and distributed web crawler are given below:

5.1.1. Apache Nutch (1.9)

It is a higher scalable and extensible open source software project of web crawler. Nutch can code entire in java programming language, but information is written into language-independent formats. It is high modular architecture, allow to developers for creating plug-ins to data retrieval, media-type parsing, clustering and querying [65]. The fig.5.1 shows the downloading of URLs using Nutch source version in eclipse.

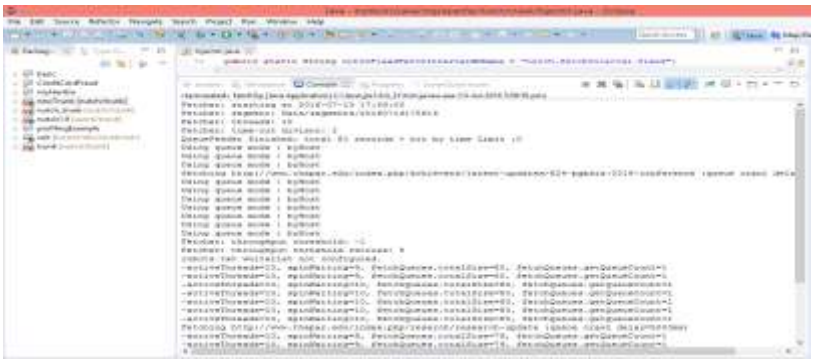


Fig.5.1: Downloading of URLs using Nutch source version in eclipse

5.1.2. Apache Hadoop-Core (0.20.2)

It is open-source framework software for distributed process and distributed storage of huge data sets on the computer clusters which is made from commodity hardware.

The Hadoop modules are designed with a fundamental assumption that failures of hardware are a common thing and thus are automatically handled by this framework. The Apache Hadoop Core consists of a storage part which is called as processing part and Hadoop Distributed File System called MapReduce. The base Apache Hadoop framework is composed of the following modules: Hadoop Distributed File System, Hadoop Common, Hadoop MapReduce and Hadoop YARN [66]. The fig.5.2 depicts the various hadoop jars used by the Nutch 1.9.

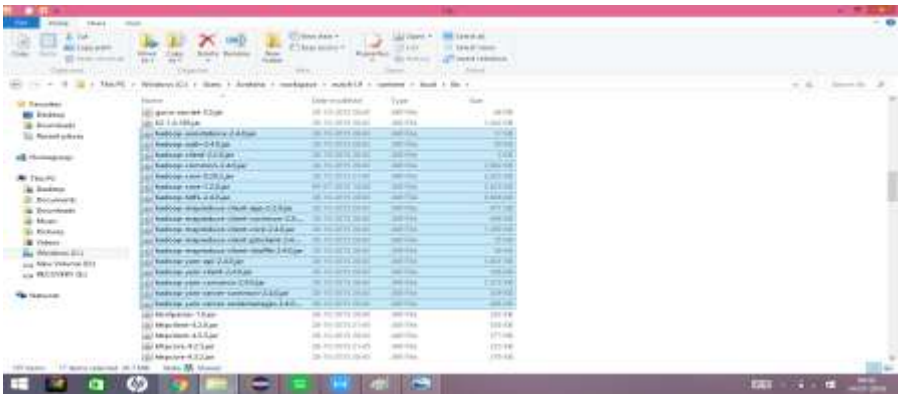


Fig.5.2: various hadoop jars used by the Nutch 1.9

5.1.3. Apache Solr 4.10.2

It is an open source search platform customized from a Java library called Lucene. The major advantage is easy-to-use search platform for enterprise equipped with functionality of indexing and searching documents and email attachments. Solr works on Hypertext Transfer Protocol (HTTP) Extensible Markup Language (XML) generally at port 8080. It also has application program interfaces (APIs) for Javascript Object Notation (JSON), Python, and Ruby [67]. In fig.5.3 one can see the solr giving results to queries passed on by nutch (and indexed by solr).

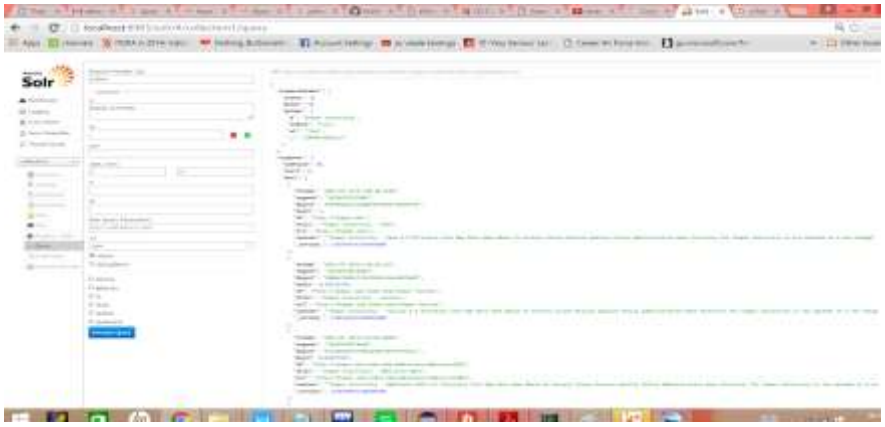


Fig.5.3: Solr giving results to queries passed on by nutch (and indexed by solr)

5.1.4. Java 1.8.0_31

Java is a programming language especially designed to comprise all the object-oriented features of C++ along with a simpler syntax also suitable for a distributed environment of the internet. Any applications are a desktop-based or remote server based (even in distributed environment). The java versions used and their JAVA_HOME path set in environment variables have been shown in fig.5.4.

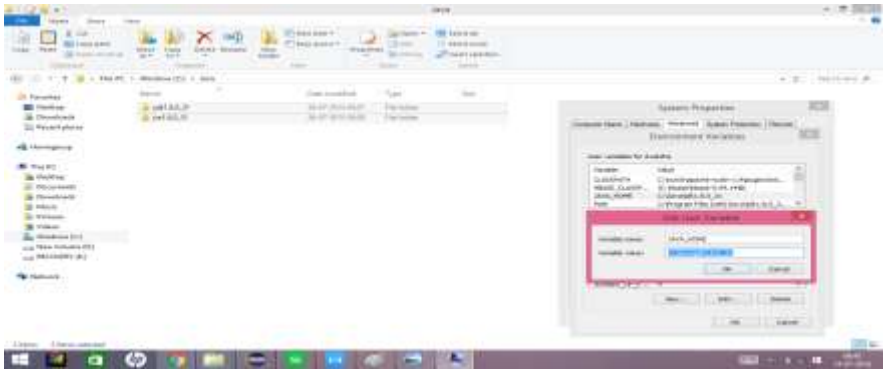


Fig.5.4: Java 1.8.0_31

5.1.5 Eclipse Luna IDE

Eclipse is an integrated development environment (IDE) used to create, debug, run, integrate and simulate many computer programming languages, especially Java. It consists of a workspace where all the projects created or downloaded are stored and there is a “run configurations” dialog box to pass input parameters and run the desired projects [68]. The basic structure of Eclipse IDE with a workspace open on it is displayed in fig.5.5.

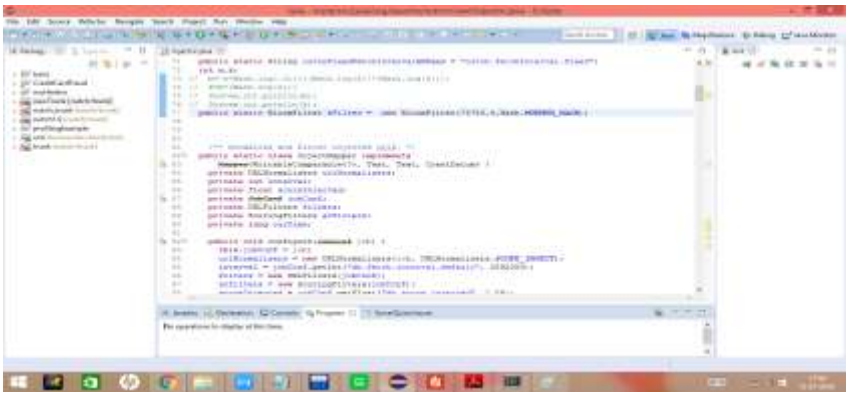


Fig.5.5: View of Eclipse IDE

5.1.6 Apache Ant 1.9.2

Apache Ant is an extension of Apache Tomcat project and is for automating software build processes. The inception of the idea came from various bugs in the UNIX, Make automated build tool and thus was implemented in Java. The major difference between Ant and Make is that Ant uses XML to describe the build process and its dependencies, while Make uses Make file format [69]. The progress of building Nutch project using Apache Ant can be viewed in fig.5.6.

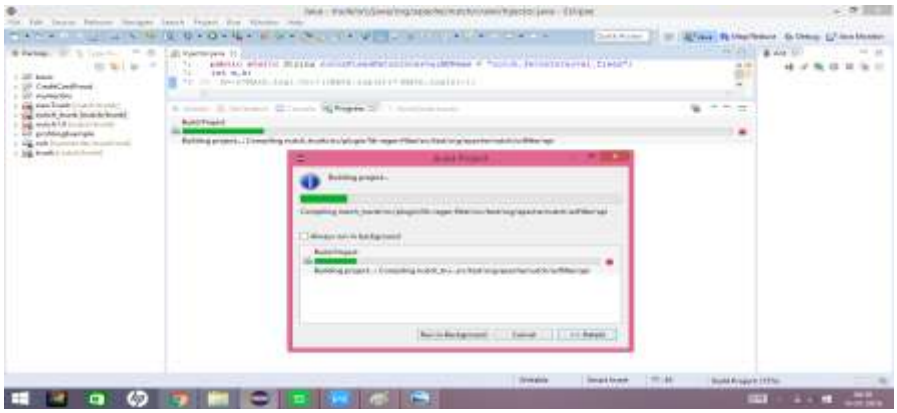


Fig.5.6: Progress of building nutch project using Apache Ant

5.1.7 IvyDe 2.2.0

Ivy is a generic dependency manager to manipulate jars or any other resources. IvyDE can also facilitate dependency management for non-Java Eclipse projects. The dependency

manager Ivy, running live to manage many jars associated with running Nutch on eclipse, is exhibited in fig.5.7.

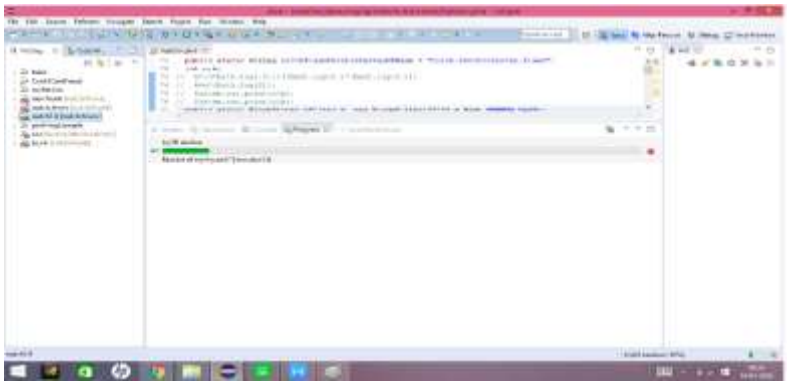


Fig.5.7: IvyDe2.2.0

5.1.8 Subclipse 1.10.9

Subclipse is an Eclipse Team Provider plug-in giving support for Subversion within the Eclipse IDE. The software is released under the Eclipse Public License (EPL) 1.0 open source license. The subclipse jars downloaded from eclipse marketplace are depicted in fig.5.8.

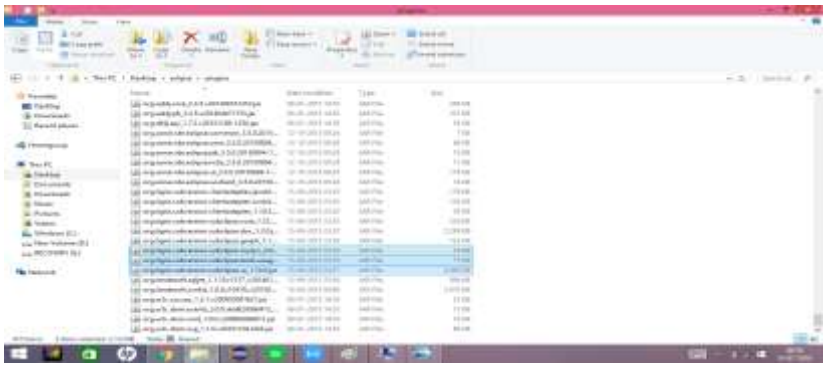


Fig.5.8: Subclipse jars downloaded from eclipse marketplace

5.1.9 Maven m2e 1.5.2

M2Eclipse supports automatic downloading of complete projects (with source files) into the eclipse workspace. M2e provides a natural and intuitive environment to use Maven with eclipse. Fig.5.9 indicates the maven m2e plugins being used in the eclipse IDE.

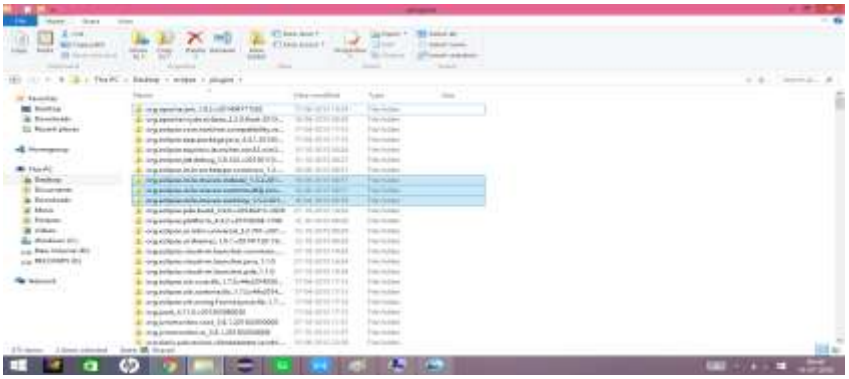


Fig.5.9: Maven m2e plugins being used in the eclipse IDE

5.1.10 Cygwin64

It a large collection of GNU and Open Source tools to provide a functionality similar to Linux distribution on Windows. It is compatible with x86 32 bit and 64 bit versions of Windows, starting with Windows XP [70]. Fig.5.10 shows how a linux-compatible project can be run easily on a pseude-linux environment called Cygwin.

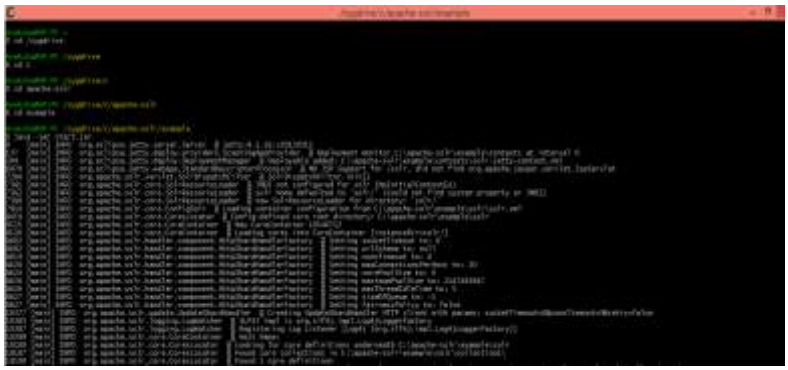


Fig.5.10: Cygwin64

5.2 Implementation:

We have implemented focused and distributed web crawler using Apache Nutch (1.9). In our experiment, we have considered four websites named as: http://www.thapar.edu, https://www.mongodb.org, https://wiki.apache.org/nutch/ and http://agrifarming.in/. We are taking four parameters named as: number of URLs crawled by focused and distributed web crawler, number of iterations taken by both crawlers, time taken by crawlers for crawling these four sites, number of hosts and type of threading taken by focused and distributed web

crawler. On the basis of these 4 parameters we are comparing focused and distributed web crawler.

Parameters:

1. Number of URLs crawled by focused and distributed web crawler:

As we can see in the table 5.1 number of URLs crawled by the distributed web crawler is more than the focused web crawler. Still, this doesn't qualify distributed as the better approach because this crawl downloads many additional URLs without adding to the relevance to the seed URLs. It diverges the crawl, causing more number of URLs to be retrieved which have no guarantee of being relevant.

Table.5.1: Number of URLs crawled by focused and distributed.

Number of URLs		
Samples	**Focused Web Crawler**	**Distributed Web Crawler**
http://www.thapar.edu	246	531
https://www.mongodb.org	16	72
https://wiki.apache.org/nutch/	106	588
http://agrifarming.in/	168	439

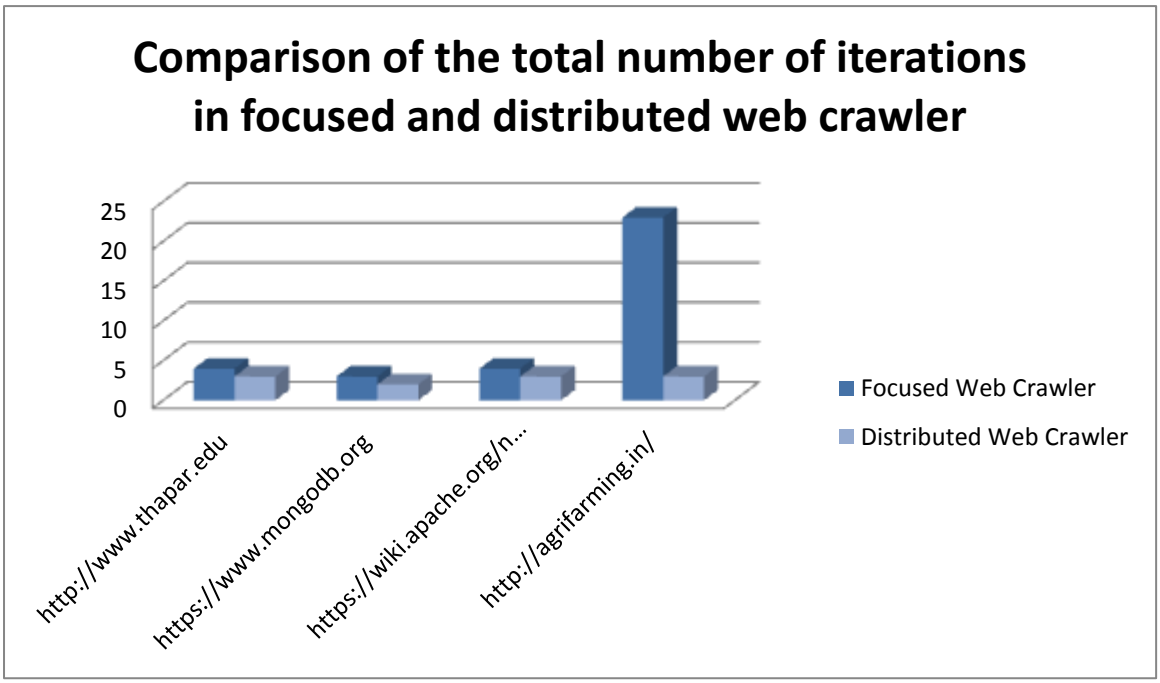
2. Comparision of the number of iterations in focused and distributed web crawler:

As shown in the table 5.2 the distributed web crawler takes lessees number of iterations and crawls more number of URL's for a given sample than the focused crawler in this lessees iteration. So, intuitively distributed crawler seems to be a better choice due retrieval of more quantity of URL's however the key is the quality and quantity of URL's. A distributed crawler is least bothered about the relevance of URL's here in our experiments relevance is measured by adherence host specified in seed URL's. A distributed crawler is unbiased and thus downloads all the URL's. It encounters through the anchor text so, in lesser number of iterations, it can crawl more URL's. However, a focused is biased toward the URL's from the same host and hence due to being very selective about the URL's. It

chooses, the crawl is convergent and retrieves lesser number of URL's even in more number of Iteration.

Table 5.2: Comparison of the total number of iterations in focused and distributed web crawler.

Number of iterations		
Samples	**Focused Web Crawler**	**Distributed Web Crawler**
http://www.thapar.edu	4	3
https://www.mongodb.org	3	2
https://wiki.apache.org/nutch/	4	3
http://agrifarming.in/	23	3



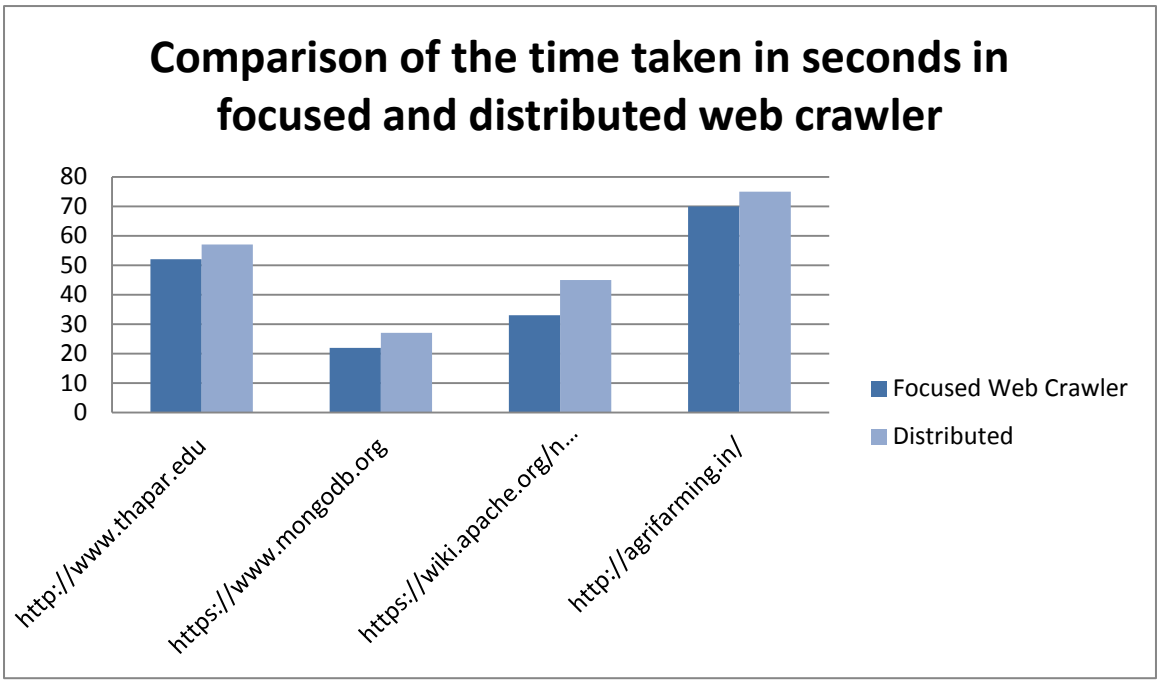
3. Comparison of the time taken in seconds in focused and distributed web crawler:

The second parameter to compare both the crawling techniques is the time as shown in the table 5.3. One may initially argue that since distributed takes lesser number of iterations to retrieve more URLs. It also takes lesser time on the contrary despite taking fewer numbers of iterations, a distributed crawler take small overall time in the entire crawl process.

It is because generate fetches and parses more number of URLs and hence updates database of discovered URLs with an enormous amount of URLs. This takes the crawl so divergent that if not stopped manually specifying the crawl depth, the distributed crawling may go on virtually for an infinite times. This is not the case with a focused crawler which believes in generating, fetching, parsing only the selected few relevant URLs, in turn updating the database of discovered URLs only with URLs which has relevant (out of all the parsed anchor text). Hence it can take some time only in the update face to choose relevant URLs. In all other faces, it is least burdened with enormous quality of irrelevant URLs. Due to its decision making capability a focused crawler finally cuts on time taken for crawl and in today's scenario time taken to do a task is one of the most important parameter to judge an application performance.

Table 5.3: Comparison of the time taken in seconds in focused and distributed web crawler

Time taken in seconds		
Samples	**Focused Web Crawler**	**Distributed Web Crawler**
http://www.thapar.edu	52	57
https://www.mongodb.org	22	27
https://wiki.apache.org/nutch/	33	45
http://agrifarming.in/	70	75



4. Comparison of the number of hosts and type of threading in focused and distributed web crawler:

Table 5.4 shows that only single host is used in focused web crawler and various hosts are used in distributed web crawler. So, focused web crawler is single threaded and distributed web crawler is multithreaded.

Table 5.4: Comparison of the number of hosts and type of threading in focused and distributed web crawler

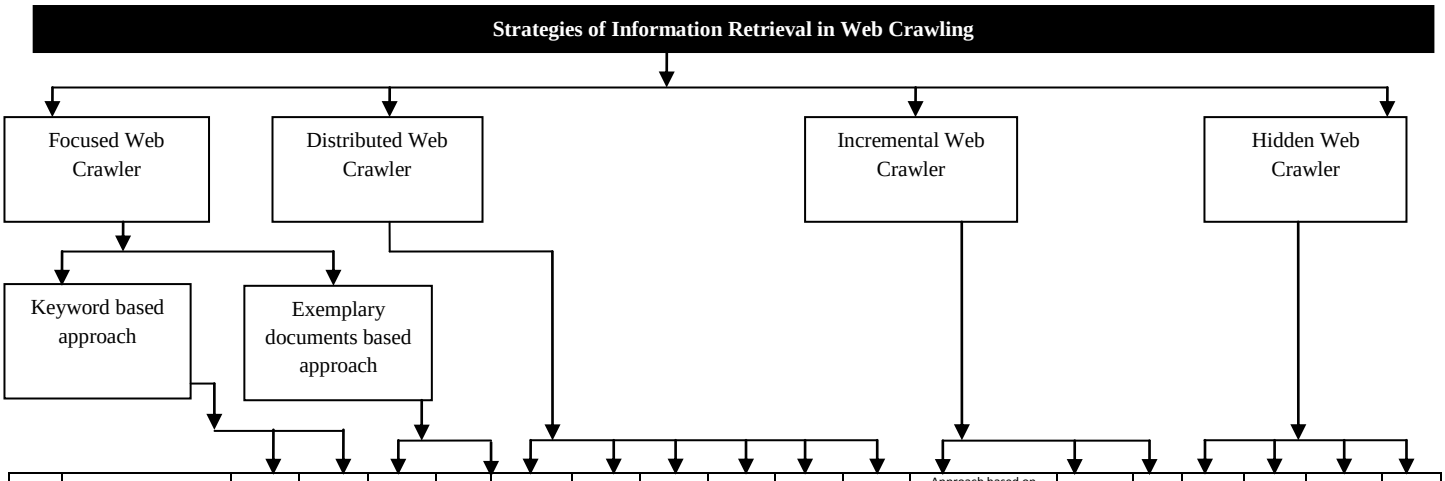
Number of hosts and type of threading	
Focused Web Crawler	**Distributed Web Crawler**
1	32
1	8
1	132
1	2
Single threaded	**Multithreaded**

Chapter 6

Conclusion and Future Scope

From the given literature and experimental setup, it has been concluded that each and every methodology of information retrieval given above support some unique features/parameters which help in depicting various relationships.

The four different strategies of information retrieval in a web crawling have been compared on the basis of 21 parameters as shown in fig.6.1. Further the two of these strategies (focused and distributed web crawler) have been implemented and various experiments have been conducted. The authors after various experiments ascertained on the basis of 4 experimental parameters that focused crawler is indeed better web the naïve implementation of a distributed crawler the drawback of focused crawler is its too convergent nature while a distributed crawler is very divergent. The best qualities of both crawlers: ability to discover URLs from many domains in just few iterations (in this distributed crawler) and the decision making ability to crawl only the relevant data (thus taking lesser time), can be combined to form a crawler with the better relevance major (such as capability to deicide the URLs related to the seed URLs) and divergent of discovery various hosts for particular need.



Sr. No.	Parameterss	Ontology based approach[15,20,43]	Link Semantic based approach [16]	Data mining based approach [18,19,21,22,25,38]	Other misc approaches [17,37,44,45]	Map reduce based approach [11,42]	Scalability based approach [3,5,41]	P2P and DHT based approach [6,39,46]	Model based approach [4,40]	Data mining based approach [13]	Other misc approaches [1,2]	Approach based on improving the freshness using calculation of refresh time of pages [7,9,24]	Data mining based approach [8,10,12,14,47]	Others misc approaches [23]	Tree based approach [49,50]	Security based approach [27]	Domain Specific based approach [29,30,32,34,35,36]	Other misc approaches [26,28,31,33,48]
1.	Scalability in terms of crawling speed.	N	N	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y	Y
2.	Robustness	N	N	N	Y	Y	N	N	N	N	Y	N	Y	N	N	Y	N	N
3.	Utilization of network resources.	N	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	Y
4.	Performance	Y	N	Y	Y	Y	Y	Y	Y	N	Y	N	Y	Y	Y	Y	Y	Y
5.	Crawling efficiency and effectiveness.	Y	Y	Y	Y	N	Y	Y	Y	N	Y	Y	Y	Y	Y	Y	Y	Y
6.	Network bandwidth.	N	N	Y	Y	Y	Y	Y	N	N	N	N	Y	Y	N	Y	N	N
7.	Reliability	N	N	N	N	Y	Y	Y	Y	N	N	N	Y	N	N	N	N	N
8.	Synchronization	N	Y	N	N	Y	Y	Y	Y	N	N	N	N	N	N	Y	N	N
9.	Load balancing.	N	Y	N	Y	N	N	Y	Y	Y	N	N	Y	Y	N	Y	N	N
10.	Crawl speed.	N	N	Y	Y	N	N	Y	Y	Y	N	N	Y	Y	N	Y	N	N
11.	To reduce the network traffic.	N	Y	N	N	Y	N	N	Y	N	N	Y	N	Y	N	Y	N	N
12.	Local connection fresh.	N	N	Y	N	N	N	Y	N	Y	N	Y	Y	N	N	N	Y	Y
13.	Up-to-date repository.	N	Y	N	Y	Y	N	Y	N	Y	N	Y	Y	N	N	N	Y	Y
14.	Harvest rate.	N	N	Y	N	N	N	N	N	N	N	N	N	Y	N	Y	N	N
15.	Maintenance of consistency.	Y	N	Y	N	Y	Y	Y	N	Y	N	N	N	Y	N	Y	Y	N
16.	Flexibility.	Y	N	Y	Y	N	Y	N	Y	Y	Y	N	Y	Y	N	N	N	Y
17.	High adaptability.	N	Y	Y	N	N	N	N	Y	N	N	N	Y	N	Y	N	Y	Y
18.	Relevant results.	Y	Y	Y	Y	Y	Y	N	N	N	N	N	Y	Y	N	Y	Y	Y
19.	Accurate and quantitative data extraction.	Y	N	Y	Y	N	Y	N	N	N	N	N	N	N	Y	N	Y	N
20.	Fault tolerance.	N	N	N	N	N	Y	Y	Y	N	N	N	N	Y	N	Y	Y	N
21.	Less download time.	Y	N	Y	Y	Y	Y	Y	Y	Y	N	Y	N	Y	Y	Y	Y	Y

Fig.6.1: Comparison of methodologies of information retrieval in web crawling

Strategies included in information retrieval for web crawling have been discussed in this literature exhaustively. In a commercial search engine a crawler should have incremental ability to scale seamlessly in a distributed environment with the use of machine leaning and focused crawling techniques to provide personalized user experience. The ultimate goal will be a robust crawler which can retrieve information not only from the hyperlinks but also from

the hidden databases of various web servers so studying and implementing a crawler with myriad capability qualify as a future scope for this work.

Moreover, a robust crawler having the communicative capabilities of all the four strategies could be commutative implemented and its performance could be compared to these four strategies running in parallel but, individually.

References

[1] "Web Crawling", http://infolab.stanford.edu/~olston/publications/crawling_survey.pdf [as accessed on 10/1/2016].

[2] "Web Crawler: A Review", http://www.ijcsms.com/journals/Volume%2012,%20Issue%2001,%20January%202012_WEB_CRAWLER1.pdf [as accessed on 2/2/2016].

[3] "Focused Web Crawler", http://www.ipcsit.com/vol45/029-ICIKM2012-M0065.pdf [as accessed on 4/2/2016].

[4] "A Simple Focused Crawler", http://cs.brynmawr.edu/Courses/cs380/fall2006/wwww12DanielePoster.pdf [as accessed on 5/2/2016].

[5] "Focused Web Crawling: A Generic Framework for Specifying the User Interest and for Adaptive Crawling Strategies", http://www.cs.sfu.ca/~ester/papers/vldb2001.pdf [as accessed on 9/2/2016].

[6] "Improving the Performance of Focused Web Crawlers", http://www.intelligence.tuc.gr/~petrakis/publications/BaPeMi09.pdf [as accessed on 16/1/2016].

[7] "A Study of Focused Web Crawlers for Semantic Web", http://www.ijcsit.com/docs/Volume%204/vol4Issue3/ijcsit2013040301.pdf [as accessed on 20/1/2016].

[8] "Incremental Crawling", http://static.googleusercontent.com/media/research.google.com/en//pubs/archive/34403.pdf [as accessed on 23/1/2016].

[9] "Study of Indexing Techniques to Improve the Performance of Information Retrieval in Telugu Language", http://www.ijetae.com/files/Volume3Issue1/IJETAE_0113_76.pdf [as accessed on 23/2/2016].

[10] "Incremental crawling with Heritrix", http://iwaw.europarchive.org/05/papers/iwaw05-sigurdsson.pdf [as accessed on 21/1/2016].

[11] "DESIGN OF A NOVEL INCREMENTAL PARALLEL WEBCRAWLER", http://www.jiit.ac.in/uploads/Final-Synopsis%20Divakar.pdf [as accessed on 22/2/2016].

[12] "Automated Indexing: The Key to Information Retrieval in the 21st century", http://www.webpages.uidaho.edu/~mbolin/obaseki.html [as accessed on 26/1/2016].

[13] "Web Crawling", http://www.itportal.in/2012/02/what-is-web-crawler-information.html [as accessed on 12/2/2016].

[14] "CrawlWave: A Distributed Crawler", http://www.aueb.gr/Users/sideri/papers/crawlwave.pdf [as accessed on 13/1/2016].

[15] Gunjan Agre and Snehlata Dongre, "A Keyword Focused Web Crawler Using Domain Engineering and Ontology", *International Journal of Advanced Research in Computer and Communication Engineering*, Vol. 4(3), pp. 463-465, Mar 2015.

[16] Gunjan H. Agre and Nikita V.Mahaja, "Keyword focused web crawler", *2nd International Conference on Electronics and Communication Systems (ICECS)*, Coimbatore, IEEE, pp. 1089-1092, 26-27 Feb 2015.

[17] Shubham Age, Tushar Indorkar, Shital Kokate and Manisha Shitole, "A Self Adaptive Semantic Focused Web Crawler", *International Journal of Research In Science & Engineering*, Vol. 1(6), pp. 74-79.

[18] Meiyappan Yuvarani, N.Ch.S.N. Iyengar and A. Kannan , "LSCrawler: a framework for an enhanced focused web crawler based on link semantics", *IEEE/WIC/ACM International Conference on Web Intelligence (WI 2006 Main Conference Proceedings)(WI'06)*, Hong Kong, IEEE, pp. 794-800, Dec 2006.

[19] Duygu Taylan, Mitat Poyraz, Selim Akyokuş and Murat Can Ganiz, "Intelligent focused crawler: learning which links to crawl", *International Symposium on Innovations in Intelligent Systems and Applications (INISTA)*, Istanbul, IEEE , pp. 504-508 , 15-18 Jun 2011.

[20] Qingyang Xu and Wanli Zuo, "First-order focused crawling", *In Proceedings of the 16th international conference on World Wide Web*, New York, ACM, pp. 1159-1160, 8–12 May 2007.

[21] Qiang Zhu, "An algorithm OFC for the focused web crawler" , *International Conference on Machine Learning and Cybernetics*, Hong Kong, IEEE, Vol. 7, pp. 4059-4063, 19-22 Aug 2007.

[22] S. Mali and B.B. Meshram, "Focused web crawler with revisit policy", *In Proceedings of the International Conference & Workshop on Emerging Trends in Technology*, New York, ACM, pp. 474-479, 25-26 Feb 2011.

[23] Soumen Chakrabarti, Martin van den Berg and Byron Dom, "Focused crawling: a new approach to topic-specific Web resource discovery", *Computer Networks*, Vol. 31(11), pp. 1623-1640, 17 May 1999.

[24] Mejdl S. Safran, Abdullah Althagafi and Dunren Che, "Improving relevance prediction for focused Web crawlers", *11th International Conference on*

Computer and Information Science (ICIS), Shanghai, IEEE, pp. 161-166, May 30 2012-Jun 1 2012.

[25] Anish Gupta and Priya Anand, "Focused web crawlers and its approaches", *International Conference in Futuristic Trends on Computational Analysis and Knowledge Management (ABLAZE)*, Noida, IEEE, pp. 619-622, 25-27 Feb 2015.

[26] Dvijesh Bhatt, Daiwat Amit Vyas and Sharnil Pandya, "Focused Web Crawler", *Advances in Computer Science and Information Technology (ACSIT)*, Vol. 2(11), pp. 1-6, Apr-Jun 2015.

[27] Hardik P. Trivedi, Gaurav N. Daxini, Jignesh A. Oswal, Vinay D. Gor and Swati Mali, "An Approach to Design Personalized Focused Crawler", *International Journal of Computer Science and Engineering*, Vol. 2(3), pp. 144-147, 31 Mar 2014.

[28] Pranali Kale and Prashant Dahiwale, "Design and Implementation of Focused Web Crawler Using Genetic Algorithm", *International Journal of Advanced Engineering and Global Technology*, Vol. 3(11), pp. 1392-1398, Dec 2015.

[29] Do Le Quoc, Christof Fetzer, Pascal Felber, Etienne Riviere, Valerio Schiavoni and Pierre Sutra, "UniCrawl: A Practical Geographically Distributed Web Crawler", *8th International Conference on Cloud Computing*, New York City, NY, IEEE, pp. 389-396, Jun 27 2015-Jul 2 2015.

[30] Poonam Ghuli and Rajashree Shettar, "A novel approach to implement a shop bot on distributed web crawler", *In Advance Computing Conference (IACC)*, Gurgaon, IEEE, pp. 882-886, 21-22 Feb 2014.

[31] Jose Exposto, Joaquim Macedo, Antonio Pina, Albano Alves and Jose Rufino, "Geographical partition for distributed web crawling", *In Proceedings of the*

workshop on Geographic information retrieval, New York, ACM, pp. 55-60, 4 Nov 2005.

[32] Milly Kc, Markus Hagenbuchner and Ah Chung Tsoi, "A scalable lightweight distributed crawler for crawling with limited resources", *International Conference on Web Intelligence and Intelligent Agent Technology*, Washington, IEEE, pp. 663-666, 2008.

[33] M. Sunil Kumar and P. Neelima, "Design and implementation of scalable, fully distributed web crawler for a web search engine", *International Journal of Computer Applications (0975-8887)*, Vol. 15(7), pp. 8-13, Feb 2011.

[34] P. Boldi, S. Vigna, B. Codenotti and M. Santini, "Trovatore: Towards a Highly Scalable Distributed Web Crawler", WWW Posters, pp. 1-2, 2001.

[35] Fei Liu , Ma Fan-yuan , Ye Yun-ming , Ming-lu Li and Jia-di Yu, "IGLOOG: A distributed Web crawler based on grid service", *7th Asia-Pacific Web Conference on Web Technologies Research and Development-APWeb*, Shanghai, Springer Berlin Heidelberg, pp. 207-216, 2005.

[36] Anup A Garje, Prof. Bhavesh Patel and Dr. B. B. Meshram, "Realizing Peer-to-Peer and Distributed Web Crawler", *International Journal of Advanced Research in Computer Engineering & Technology*, Vol. 1(4), pp. 353-357, Jun 2012.

[37] Boon Thau Loo, Owen Cooper and Sailesh Krishnamurthy, "Distributed web crawling over DHTs", Feb 2004.

[38] Seyed M. Mirtaheri, Di Zou, Gregor von Bochmann, Guy-Vincent Jourdan and Iosif-Viorel Onut, " Dist-ria crawler: A distributed crawler for rich internet applications", *Eighth International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC)*, Compiegne, IEEE, pp. 105-112, 28-30 Oct 2013.

[39] Mitra Nasri, Saeed Shariati and Mohsen Sharifi, “Availability and Accuracy of Distributed Web Crawlers: A Model-Based Evaluation”, *Second UKSIM European Symposium on Computer Modeling and Simulation*, Liverpool, IEEE, pp. 453-458, 8-10 Sept 2008.

[40] Vladislav Shkapenyuk and Torsten Suel, “Design and implementation of a high-performance distributed web crawler”, *Proceedings 18th International Conference on Data Engineering*, San Jose, IEEE, pp. 357-368, 26 Feb 2002-01 Mar 2002.

[41] Bing Zhou, Bo Xiao, Zhiqing Lin and Chuang Zhang, “A distributed vertical crawler using crawling-period based strategy”, *2nd International Conference on Future Computer and Communication (ICFCC)*, Wuhan, IEEE, pp. V1-306 - V1-311, 21-24 May 2010.

[42] Pallavi and Rajiv Sharma, “An Adaptive, Selective and Incremental Web Crawler”, *International Journal of Scientific Engineering and Research (IJSER)*, Vol. 3(7), pp. s198-202, Jul 2015.

[43] Hadrien Bullot, Shyam K. Gupta and Mukesh K. Mohania, "A data-mining approach for optimizing performance of an incremental crawler", *International Conference on Web Intelligence*, IEEE, pp. 610-615, 2003.

[44] Jenny Edwards, Kevin McCurley and John Tomlin, "An adaptive model for optimizing performance of an incremental web crawler", *In Proceedings of the 10th international conference on World Wide Web*, New York, ACM, pp. 106-113, May 2001.

[45] Tan Qingzhao Tan and Prasenjit Mitra, "Clustering-based incremental web crawling", *ACM Transactions on Information Systems (TOIS)*, Vol. 28(4),ACM, pp. 1-26, Nov 2010.

[46] Christos Bouras, Vassilis Poulopoulos and Athena Thanou, "Creating a polite adaptive and selective incremental crawler", *In IADIS International Conference WWW/INTERNET*, pp. 307-314, 2005.

[47] Niraj Singhal, Ashutosh Dixit and Dr. A. K. Sharma, "Design of a Priority Based Frequency Regulated Incremental Crawler", *International Journal of Computer Applications*, Vol. 1(1), pp. 47-52, 2010.

[48] A.K. Sharma and Ashutosh Dixit, "Self adjusting Refresh Time based Architecture for incremental web crawler", *International Journal of Computer Science and Network Security*, Vol. 8(12), pp. 349-354, Dec 2008.

[49] Junghoo Cho and Hector Garcia-Molina, "The evolution of the web and implications for an incremental crawler", pp. 1-18, 2 Dec 1999.

[50] Jiang-Ming Yang, Rui Cai, Chunsong Wang, Hua Huang, Lei Zhang and Wei-Ying Ma, "Incorporating site-level knowledge for incremental crawling of web forums: A list-wise strategy", *In Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, New York, ACM, pp. 1375-1384, Jun 28–Jul 1 2009.

[51] Weicheng Ma, Xiuxia Chen and Wenqian Shang, "Advanced deep web crawler based on Dom", *Fifth International Joint Conference on Computational Sciences and Optimization (CSO)*, Harbin, IEEE, pp. 605-609, 23-26 June 2012.

[52] R. Anita, V. Ganga Bharani, N. Nityanandam and Pradeep Kumar Sahoo, "Deep iCrawl: An Intelligent Vision-Based Deep Web Crawler", *International Journal of Computer, Electrical, Automation, Control and Information Engineering*, Vol. 5(2), pp. 128-133, 2011.

[53] Manvi, Ashutosh Dixit, Komal Kumar Bhatia and Jyoti Yadav, "Design and Implementation of Domain based Semantic Hidden Web Crawler", *International*

Journal of Innovations & Advancement in Computer Science IJIACS, Vol. 4, pp. 73-84, May 2015.

[54] Rosy Madaan, Ashutosh Dixit, A.K. Sharma and Komal Kumar Bhatia, "A framework for incremental hidden web crawler", *International Journal on Computer Science and Engineering*, Vol. 2(3), pp. 753-758, 2010.

[55] Nupur Gupta and Shalini Kapoor, "Extraction of Query Interfaces for Domain-Specific Hidden Web Crawler", *International Journal of Computer Science and Information Technologies (IJCSIT)*, Vol. 5(1), pp. 679-681, 2014.

[56] Ali I. El-Desouky, Hesham A. Ali and Sally M. El-Ghamrawy, "An Automatic Label Extraction Technique for Domain-Specific Hidden Web Crawling (LEHW)", *The 2006 International Conference on Computer Engineering and Systems*, Cairo, IEEE, pp. 454-459, 5-7 Nov 2006.

[57] Dilip Kumar Sharma and A. K. Sharma, "A qiiiep based domain specific hidden web crawler", *In Proceedings of the International Conference & Workshop on Emerging Trends in Technology*, New York, ACM, pp. 224-227, Feb 25–26 2011.

[58] Komal Kumar Bhatia, A.K. Sharma and Rosy Madaan, "AKSHR: A novel framework for a Domain-specific Hidden Web Crawler", *1st International Conference on Parallel Distributed and Grid Computing (PDGC)*, Solan, IEEE, pp. 307-312, 28-30 Oct 2010.

[59] Xin Wang, Luhua Wang, Gengyu Wei, Dongmei Zhang and Yixian Yang, "Hidden web crawling for SQL injection detection", *3rd IEEE International Conference on Broadband Network and Multimedia Technology (IC-BNMT)*, Beijing, IEEE, pp. 14-18, 26-28 Oct 2010.

[60] Sonali Gupta and Komal Kumar Bhatia, "A comparative study of hidden web crawlers", *International Journal of Computer Trends and Technology (IJCTT)*, Vol. 12(3), pp. 111-118, Jun 2014.

[61] Satinder Bal Gupta, "Challenges in Designing a Hidden Web Crawler", *International Journal of Information Technology & Systems*, Vol. 2(1), pp. 117-122, Jun 2013.

[62] Karane Vieira, Luciano Barbosa, Juliana Freire and Altigran Silva, "Siphon++: a hidden-webcrawler for keyword-based interfaces", *In Proceedings of the 17th ACM conference on Information and knowledge management*, New York, ACM, pp. 1361-1362, Oct 26–30 2008.

[63] Smita Agrawal and Kriti Agrawal, "Deep Web Crawler: A Review", *International Journal of Innovative Research in Computer Science & Technology (IJIRCST)*, Vol. 1(1), pp. 12-15, Sept 2013.

[64] Jihwan Song, Dong-Hoon Choi and Yoon-Joon Lee, "OGSA-DWC: A Middleware for Deep Web Crawling Using the Grid", *Fourth International Conference on eScience, 2008. eScience'08.* Indianapolis, IEEE, pp. 370-371, 7-12 Dec 2008.

[65] "Crawling the Hidden Web", http://www.dia.uniroma3.it/~vldbproc/017_129.pdf [as accessed on 2/1/2016].

[66] "Crawling the Content Hidden Behind Web Forms", http://www.tic.udc.es/~mad/publications/cchiddenbwf_extended.pdf [as accessed on 5/1/2016].

[67] "Web Crawler: Extracting the Web Data", http://www.ijcttjournal.org/Volume13/number-3/IJCTT-V13P128.pdf [as accessed on 11/1/2016].

[68] “A Web Crawler System Design Based on Distributed Technology”, http://ojs.academypublisher.com/index.php/jnw/article/viewFile/jnw061216821689/4020 [as accessed on 25/1/2016].

[69] “Web Crawling and IR”, http://www.cfilt.iitb.ac.in/resources/surveys/CrawlingAndIR-Varun-May14.pdf [as accessed on 22/2/2016].

[70] “Indexing in Information Retrieval”,http://www.dc.fi.udc.es/~roi/publications/rblanco-phd.pdf [as accessed on 25/1/2016].

Annexure I

List of Publications

Accepted Paper:

[1] Chandni Saini and Vinay Arora, "**Information retrieval in web crawling: A Survey**" in the 5th International Conference on Advances in Computing, Communications and Informatics, Jaipur, IEEE, 2016.

Annexure II

Video Link

https://www.youtube.com/channel/UCKx5fYOoxg_SbvLA-qQ5AKg

Annexure III

Plagiarism Certificate

aaa

ORIGINALITY REPORT

10%	7%	4%	4%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Panjab University Student Paper	1%
2	Submitted to ABV-Indian Institute of Information Technology and Management Gwalior Student Paper	1%
3	files.figshare.com Internet Source	<1%