A HYBRID PAGE RANKING ALGORITHM

Dissertation submitted in

partial fulfilment of the requirement

for the award of the degree of

Master of Technology

in

Computer Science and Engineering

by

UNIQUE KUMAR KATNORIA
University Roll No. 2K12/CSE/23

Under the Esteemed Guidance of

Mr. Manoj Kumar
Associate Professor, Computer Engineering Department, DTU

2012-2014

COMPUTER ENGINEERING DEPARTMENT

DELHI TECHNOLOGICAL UNIVERSITY,

DELHI – 110042, INDIA

CERTIFICATE

This is to certify that the dissertation titled " **A HYBRID PAGE RANKING ALGORITHM** " is a bonafide record of work done at Delhi Technological University by **UNIQUE KUMAR KATNORIA**, Roll No. **2K12/CSE/23** for partial fulfilment of the requirements for the degree of Master of Technology in Computer Science & Engineering. This project was carried out under my supervision and has not been submitted elsewhere, for the award of any other degree or diploma to the best of my knowledge and belief.

(Mr. Manoj Kumar)
Associate Professor
Department of Computer Engineering
Delhi Technological University

Dated

ACKNOWLEDGEMENT

First of all, I would like to express my deep sense of respect and gratitude to my project supervisor Mr. Manoj Kumar for providing the opportunity of carrying out this project and being the guiding force behind this work. I am deeply indebted to him for the support, advice and encouragement he provided without which the project could not have been a success.

Secondly, I am grateful to Dr. O.P Verma, HOD, Computer Engineering Department, DTU for his immense support. I would also like to acknowledge Delhi Technological University for providing the right academic resources and environment for this work to be carried out.

Last but not the least I would like to express sincere gratitude to my parents and friends for constantly encouraging me during the completion of work.

UNIQUE KUMAR KATNORIA
University Roll no: 2K12/CSE/23
M.Tech (Computer Science & Engineering)
Department of Computer Engineering
Delhi Technological University
Delhi – 110042

Abstract

The web has now become a very huge network. With the ever increasing pages and users on the web it has become very enormous. The challenge by this is that the ordering of information has become a very difficult task.

The modern user queries the Internet. The user needs the information. It uses a search engine for this purpose. The task of the search engine is to answer the user query in very precise and fast manner.

There have been very large number of algorithm for ranking the documents on the web. The most popular of them are PageRank, RatioRank, INDEGREE, Salsa etc. But the modern day web is even challenging these well-known processes. The web has become a very difficult to manage information.

Therefore, we propose a algorithm for processing the information. The algorithm uses content based segmentation to find the occurrences of term in the page. The value given by the processes is used as the initial value for each page's rank in the ranking algorithm. The ranking algorithm is made to distribute the value of page among its hyperlinks. Also, we use the differential normalization technique to decrease the number of iterations done by the algorithm. The model was implemented and the results were found out be very good. The system is particularly useful for peer – to peer networks peer or domain based searching which have a small number of nodes, but with better computing environment it can be easily scaled to the web.

Keywords – search engine, pagerank, HITS, PHITS, Indegree, proportional rank distribution, content- based block segmentation, differential normalized ranking.

Table Of Contents

List Of Figures

List Of Tables

1. INTRODUCTION

Nowadays, the Internet has become a major source of information on the www. More and more users are adding to Internet everyday. Also the number of webpages is increasing everyday. Also the traffic on the Internet is increasing everyday. Let alone the number of pages websites are about to become 1 billion. Also, the new users are not experienced and cannot frame their queries correctly. The number of internet users are about 2.5 billion .This makes the task of information retrieval very difficult [1]. There are about 3 billion queries to google in a single day. You have to retrieve a large set of pages for every user query. It is practically impossible for the user to access every page by itself. Here the search engine performs the task. The user enters query into the search engine and the search engine gives the desired results. This helps the user to surf the net in a fast and effective way.

But the task of search engine is very difficult. It has to collect the relevant results. Also, another major task is to order those large numbers of results in a relevant ways so as more closer results should be higher in the list. This is called raking of the results. This is the most cognitive task in search engine. This also very time consuming. So the search engines prepare index of the most popular queries so the results can be given quickly to the user. The retriever performs the task of actually getting the desired pages.

The major components of a search engine are-

1. The WebCrawler - A web crawler is automated computer programs that traverses the www for finding the web pages .Generally, a WebCrawler is run periodically and set of relevant pages for search topics are indexed. The indexed information is then used to give answers to the user queries. Two common techniques used to crawl the WWW are –

1. Breadth-first search – In this the links are stored in the queue and are traversed in the order.

2. Depth-first crawling – In this the links are traversed as soon as it is found and stacks are used.

The variables that affected crawling are -

Included pages

Most search engines will find information by beginning at one page and then following all of the links on that page. It will then follow all of the links on these new pages, and so on. Therefore, if a page is not linked to from another page, it may never be found by a search engine. Authors can include unlinked pages in a search engine by submitting them to each specific search engine.

Excluded pages

Some web administrators may choose to exclude their pages from search engines because they are internal pages or Intranets. Many web pages are also excluded because their content is dynamically generated from a database and a search engine cannot find it.

Documents types

Different search engines will search different document types. All will search HTML documents, but some will also search PDF, PowerPoint, Word, Excel, and more.

Frequency of crawling

An important part of a web crawler is how frequently it retrieves information from pages. Some sites it will visit more often than others.

2. The Database

The database is the part of WebCrawler that stores the data. It stores all the pages accessed by the WebCrawler. The user searches the database rather than the internet when it enters the query.

Proper indexing of the database improves the performance of the crawler. The variables affecting the retrieved pages are –

Data in the database

Some search engines will have extremely large while others will have comparatively small ones . A small database is not necessarily worse, however, since it may offer more focused and higher quality results than a larger database.

Updation of the database

The freshness of the database is a direct result of how frequently the web crawler retrieves new information database update is performed. If the information in the database is fairly old, then your search results will suffer.

3) The Search algorithm - The search algorithm sees the current pages to see how well they go with the input user entered query. Variables that affect the retrieved results are:

Operators

Most search engines allow you to use operators such as AND, OR, and NOT in order to create complex search statements. The terms may need to be entered in upper case.

Phrase Searching

Search engines will generally search for words as phrases when quotation marks are placed around the phrase..

Truncation

Some search engines will automatically truncate the terms you enter. This means that the search engine will not only search for the term exactly as you spelled it, but will also search on that term with alternate endings and as a plural. Some search engines will only search for variable endings on certain common words.

4. The Ranking algorithm

Each search engine interprets the terms you enter into the search box in different ways. This is the most important part-b of the search engine. It determines the quality of the search results. It requires cognitive on the part of the search engine. User accessed pages are also considered in the ranking. The variables affecting the retrieved results are -

Location and Frequency

All search engines look at the location and frequency of words in a page. If a term appears near the top of a web page, such as in the title or in the first few paragraphs of text, it is assumed that the page is more relevant than if the term is used at the bottom of the page. Pages where the words appear more frequently in relation to the other words on the page also qualifies the page as being more relevant than other web pages.

Link Analysis

This feature analyzes how pages link to each other and then uses this information to determine the "importance" of each page. If a page is linked to from a large number of other pages, then it is ranked more highly.

Clickthrough Measurement

Some search engines also use Clickthrough analysis. This means that a search engine might watch what results someone selects from a particular search, then eventually drop high-ranking pages that aren't attracting clicks, while promoting lower-ranking pages that do pull in visitors.

The working of the search engine is as follows [2] .When a user fires a query in the form of keywords on the interface of a search engine, it is retrieved by the query processor component, which after matching the query keywords with the index returns the URLs of the pages to the user. But before representing the pages to the user, some ranking mechanism (web mining) either

in back end or in front end is used by most of the search engines to make the user search navigation easier between the search results. Important pages are put on the top leaving the less important pages in the bottom of the result list. Such kind of mechanism is used by a popular search engine Google that uses the information PageRank algorithm to rank its result pages.

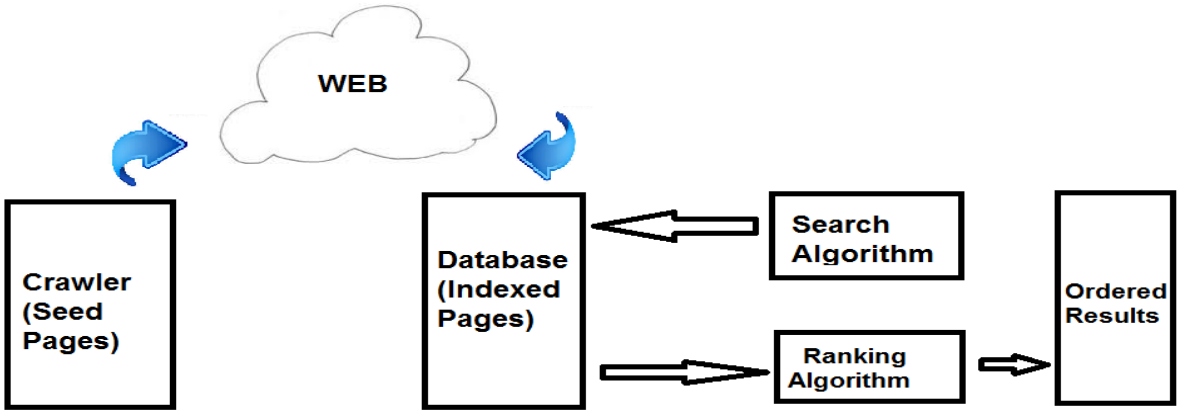


Figure 1.1 Architecture of search engine

1.1 Structure Of Web

Structure wise organization web can be seen if we consider the hyperlink design of the webpages. Each page can be considered to be a node of a graph and a hyperlink specifying a link between the nodes. This makes the web a graph. The web is directed graph is shown below.

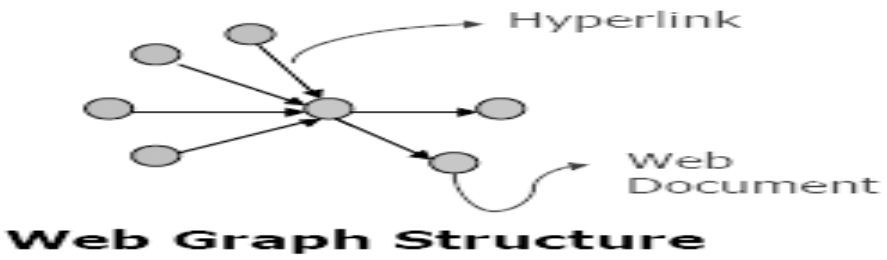


Figure 1.2 Hyper-graph representation of web

But the problem lies in the enormity of the web graph which contains billions of nodes and trillion of links. Appling searching and ordering algorithms to such a huge network are a very

difficult task and takes time in order of days. Moreover, the graph is always changing and expanding which increases the difficulty of the problem. A instance of web is shown below.

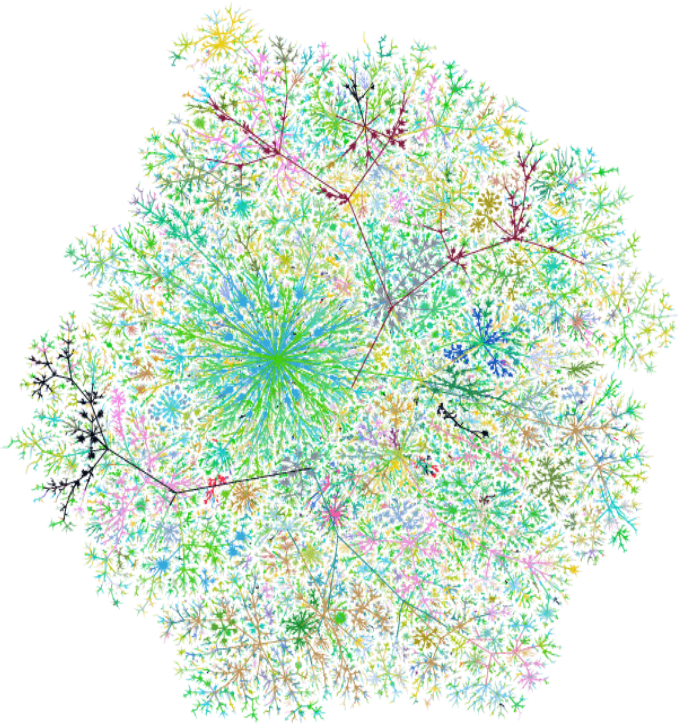


Figure 1.3 Structure of information web

As it can be seen from the structure the web is very complex and thus would require complex algorithm to traverse such difficult graph. Moreover, information retrieval would be even difficult task.

1.2. Measuring the quality of a Page

The fundamental question for search engine is how to find the importance of the page. This importance is measured with respect to the search query. Some measures tro do so are given below

1. Inlinks - The number of Inlinks pointing to the page are a good measure of information present on the page.

2. Geographical location – Some pages may be relevant for a search query coming from a specific geographical location.

3. Popularity – Some web pages are more popular and are always considered to have good information.

4. Quality - In this the quality of the Inlinks are also measured.

5. Term count – the number of times the term to be searched is found in the page.

1.3 Metrics for the Search Engine

The most important metrics for the testing of search engine are as follows –

1. Precision - Precision is measured as the number of relevant pages retrieved. It measures the overall focus of the search.

Precision= NO. of retrieved relevant pages / NO. of pages retrieved

Though in network that is so large it is impossible to find accurate measures of precision , but indirect measures are often used. Ideally, precision should be 1, but 0.6 is considered to be a good precision.

2. Recall – Accuracy is measured as the number of relevant retrieved pages out of all relevant pages. It measures the overall completeness of the search.

Recall= NO. of retrieved relevant pages/NO. of relevant pages in the set

Though in network that is so large it is impossible to find accurate measures of precision , but indirect measures are often used. Ideally, precision should be 1, but 0.6 is considered to be a good precision.

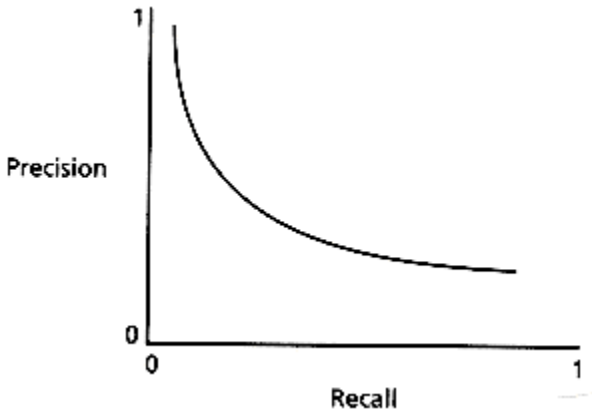


Figure 1.4 Precision and Recall

1.4 Modern approaches in ranking

Modern day ranking algorithms are very complex and take various factors into account to determine the rank of the page in the search. The factors decide the overall quality of the search engine. The various types of ranking technique used un search engine are search engines are –

1. **Link based ranking** – These are the most popular technique for ranking in web mining. In this the rank of the page is determined the links pointing to it. This technique assumes that link point to more popular pages. These are generally iterative algorithms which run until a certain condition is achieved. Almost all modern day algorithm use this technique in some way.

2. **Content based ranking** – In these technique the content contained in the page is used to determine the overall importance of the page in the search results. The content of the page are read and there is some criteria on which the score is assigned in accordance to the content in the page. The algorithms are generally faster as one page is accessed only once.

3. **User pattern based ranking** – In these techniques the previous pattern followed by the user are used to rank the results of the query. These techniques study the usage pattern of the results of the query by the user. These techniques provide a personalized experience to users. These though generally require artificial intelligence techniques for their operation.

1.5 PageRank Algorithm

We now consider the page rank algorithm. It is a link based ranking algorithm used for ranking the results of the search query. Here each page is given an initial rank and that rank is distributed among all outlinks equally. The general formula is given as –

$$PageRank\ of\ site = \sum \frac{PageRank\ of\ inbound\ link}{Number\ of\ links\ on\ that\ page}$$

OR

$$PR(u) = (1-d) + d \times \sum \frac{PR(v)}{N(v)}$$

where

pr(x) – PageRank of page x.

n(y)- number of outlinks of page y.

Σ – sum of the ranks

d – damping factor

1.6 Organization of Thesis

In chapter 1 we have introduced the concept of information retrieval (IR). In chapter 2 we will introduce some related terms. In chapter 3 we will show the literature survey. In chapter 4 we will provide the problem statement and proposed system. In chapter 5, we will show the implementation and the results of the proposed system. In chapter 6, we will provide the conclusion and the future work.

2. Related Topics

In this section n we shall study so e topics related to the concept of Information retrieval (IR). Some of the concepts covered here are WWW, Internet, Search engine, Documents, query, crawling, ranking, term frequency, inverse document frequency.

2.1 Internet

The Internet is a global system of interconnected computer networks that use the standard Internet protocol suite (TCP/IP) to link several billion devices worldwide. The Internet large number of information resources and services, such as the inter-linked hypertext documents and applications of the World Wide Web (WWW), the infrastructure to support email, and peer-to-peer networks for file sharing and internet telephony.

2.2 WWW

The World Wide Web is a subset of the Internet. The Web consists of pages that can be accessed using a Web browser. The Internet is the actual network of networks where all the information resides. The Hyper-Text Transfer Protocol (HTTP) is the method used to transfer Web pages to your computer. With hypertext, a word or phrase can contain a link to another Web site.

2.3 Search Engine

A web search engine is a software system that is designed to search for information on the World Wide

2.4 Documents

A document is well defined form of storing information which can be easily accessed. Documents are divided into different categories. A *document type* is defined by specifying the constraints which any document which is an *instance* of the type must satisfy. This helps in proper organization of information. Three types of web documents are.

Static - A static web document resides in a file that it is associated with a web server. Because the contents do not change, *each request for a static document results in exactly the same response.*

Dynamic - A dynamic web document does not exist in a predefined form. When a request arrives the web server runs an application program that creates the document. Because a fresh document is created for each request, *the contents of a dynamic document can vary from one request to another.*

Active - An active web document consists of a computer program that the server sends to the browser and that the browser must run locally..

2.5 Query

A search query is the form in which the user requests the information needed by the user. Search engines use the query to find the documents related to the query. Search engines also use query to index the web pages in response to certain kind of query.

2.6 Crawling

The search engine periodically traverses the internet to find new and updated pages and prepares the index of the pages to give good responses to the user. The part of the search engine responsible for it is called crawler.

Two common techniques used to crawl the www are –

1. **Breadth-first search** – In this the links are stored in the queue and are traversed in the order.

2. **Depth-first crawling** – In this the links are traversed as soon as it is found and stacks are used.

2.7 Ranking

When the answer to the search are provided to the search engine it has to order them in certain way so that the user can get the valid answers in the first few answers. This is the responsibility

of the ranking part of the search engine. The various ranking methods used by the search engines are –

1. **Link based ranking** – These are the most popular technique for ranking in web mining. In this the rank of the page is determined the links pointing to it. This technique assumes that link point to more popular pages. These are generally iterative algorithms which run until a certain condition is achieved.

2. **Content based ranking** – In these technique the content contained in the page is used to determine the overall importance of the page in the search results. The content of the page are read and there is some criteria on which the score is assigned in accordance to the content in the page. The algorithms are generally faster as one page is accessed only once.

3. **User pattern based ranking** – In these techniques the previous pattern followed by the user are used to rank the results of the query. These techniques study the usage pattern of the results of the query by the user. These techniques provide a personalized experience to users. These though generally require artificial intelligence techniques for their operation.

2.8 Term Frequency

Term frequency of term t in document d is defined as the number of times that t occurs in d. It measures the importance of the document in the given information.

2.9 Inverse Document frequency

Estimate the rarity of a term in the whole document collection. (If a term occurs in all the documents of the collection, its IDF is zero.)

2.10 Indexing

Search engine indexing collects, parses, and stores data to facilitate fast and accurate information retrieval. Index design incorporates interdisciplinary concepts from linguistics, cognitive psychology, mathematics, informatics, physics, and computer science.

2.10 Term-frequency Inverse document Frequency

The tf-idf weight of a term is the product of its tf weight and its idf weight.

2.11 Web Mining

Web mining is the Data Mining technique that automatically discovers or extracts the information from web documents. It consists of following tasks:

1. **Resource finding**: In this we find the resources that ma contain the desired information. Now we collect online and offline webpages

2. **Information selection and pre-processing**: This steps contains selection of information from the resources. The step involves the activities like finding the specific data, stemming and data search.

3. **Generalization**: It finds the individual pattern in a website and and across multiple websites. Pattern recognition techniques are used at this part.

4. **Analysis**: This step involves the analysis of mined pattern and find the facts from it. Pattern mining plays a important role in this step also

There are three branches of web mining according to the form input data used in the process of web mining -

Web Content Mining - It involves getting the contents of the WWW and indexing it to get a quicker access to it. It also tries to make the information in a structured form. It mainly focuses on content within the page.

Web Structure Mining – This process involves to model the link structure of the pages. The similarity and the connection between pages is analyzed using the link structure of the of the web page. This focuses on the document level connection only.

Web Usage Mining – It involves finding the usage patterns of the web users. This helps in predicting what the user is looking for by working with the secondary data on the web. This

data is collected from web servers. This data is used for analysing the requirements of the user as to what it is looking for as some might be looking for technical data, while some will be looking for entertainment data, while some might looking for financial data, while some might be looking for business communication data. This is processed using the usage structure of user.[17]

3. Literature Survey

We now present a literature survey and study the present systems.

3.1 Term- frequency and inverse document frequency

We will now examine the structure and implementation of TF-IDF for a set of documents. We will study the work in detail and look at the functioning of the topic.

Mathematical Overview

We will give a explanation of TF-IDF. Essentially, TF-IDF works by determining the relative frequency of words in a specific document compared to the inverse proportion of that word over the entire document corpus. This tells how important the word to the document is! Words that are common in a single or a small group of documents tend to have higher TFIDF numbers than common words such as articles and prepositions. The overall approach works as follows. Given a document collection D, a word w, and an individual document d ϵ D, we calculate

$$W_d = f_{w,d} * \log (|D| f_{w,D})$$

where $f_{w,\,d}$ equals the number of times w appears in d, |D| is the size of the corpus, and f_w, D equals the number of documents in which w appears in D. There are a few different situation that can occur here for each word, depending on the values of $f_{w,\,d}$, |D|, and $f_{w,\,D.}$ Assume that |D| ~ $f_{w,\,d}$, i.e. the size of the corpus is approximately equal to the frequency of w over D. If 1 < log (|D|/ $f_{w,\,d}$) < c for some very small constant c, then w_d will be smaller than f_w, d but still positive. This implies that w is relatively common over the entire corpus but still holds some importance throughout D. The TF-IDF of common word is very low which makes them very negligible. Finally, suppose $f_{w,\,d}$ is large and f_w, D is small. Then log (|D|/ $f_{w,\,D}$) will be rather large, and so w_d will likewise be large. This is the case we are most interested in, since words with high w_d imply that w is an important word in d but not common in D. This w term is said to have a large discriminatory power. Therefore, when a query contains this w, returning a document d where w_d is large will very likely satisfy the user. [3]

The survey shows that TF-IDF is an efficient and simple algorithm for matching words in a query to documents that are relevant to that query. We see that TF-IDF returns documents that are highly relevant to a particular query. If a user were to input a query for a particular topic, TF-IDF can find documents that contain relevant information on the query. Despite its strength, TF-IDF has its limitations. In terms of synonyms, notice that TF-IDF does not make the jump to the relationship between words. TF-IDF would not consider documents that might be relevant to the query but instead use the word. [3]

3.2 Crawlers

The first generations of crawlers generally used the traditional graph algorithms to traverse the Internet. A set of popular URLs are used as seed. Then the hyperlinks are followed to get a predefined number of pages. The contents of the document are not focused on. This is because the purpose is to discover the whole of the entire World Wide Web (WWW). But, the Web was very small at that time now. It is very large and so the traditional algorithms cannot be applied. So we consider alternative algorithms.[4]

Depth-first crawling searches the path till the designated end. It works by finding the first link on the first page. It then crawls the page associated with that link, finding the first link on the new page, and so on, until the end of the path has been reached. The process continues until all the branches of all the links have been exhausted. [4]

Breadth – first search stores the URLs on a page and vists them sequentially. The advantage of this technique is that the high quality results are obtained early in the procedure. This makes the next results to be of high quality. Overall rates at which the *good URLs*, as defined by human relevance judges, were no higher or lower than other label categories at any stage of the crawl. However, the disadvantage is that the progress is sub – linear in the number of unique domains crawled. [5]

Fish-Search the Web is crawled by a using a team of crawlers. If the ‘‘fish’’ finds a relevant page based on keywords specified in the query, it continues looking by following more links from that page. If the page is not relevant, its child links receive a low preferential value. [4]

Shark-Search is a more aggressive version of Fish-Search Shark-Search offers two main improvements over Fish-Search. It uses a continuous valued function for measuring relevance as opposed to the binary relevance function in Fish-Search. In addition, Shark-Search has a more refined notion of potential scores for the links in the crawl frontier.[6]

Naive Best First exploits the fact that relevant pages possibly link to other relevant pages. Therefore, the relevance of a page A to a topic T, pointed by a page B, is estimated by the relevance of page B to the topic T. After studying the various approaches in literature we find that the major open problem in focused crawling is that of properly assigning credit to all pages along a crawl route that yields a highly relevant document. Due to lack of credit assignment strategy the focused crawlers suffer from a limited ability to sacrifice short term document retrieval gains in the interest of better overall crawl performance. [4]

3.3 Types of Web crawlers

1. Focused Web Crawler

Focused Crawler is the Web crawler that tries to download pages that are related to each other. It collects stores only the pages that are relevant to the query. It is also called topic crawler. The focused crawler determines the following – relevancy, way forward. It determines how far the given page is relevant to the particular topic and how to proceed forward. The benefits of focused web crawler is that it is economically feasible in terms of hardware and network resources, it can reduce the amount of network traffic and downloads. The search exposure in the situation of focused crawling is also very large. [7]

2. Incremental Crawler

A traditional crawler, in order to refresh its collection, periodically deletes the previous pages. On the contrary, an incremental periodically refreshes the collection by visting the pages depending on how frequently they change. The less important pages are migrated by fresh pages. This resolves the problem of the access to fresh page. The benfit of increamental crawling is that it only provides useful data to users and thus the bandwidth is utilized and enrichment of user results is achieved. [7]

3. Distributed Crawler

Distributed web crawling uses distributed computing technique. Many crawlers try to distribute the computing so as to cover most of a web. The centralized server maintains the the communication among the nodes. The ranking algorithm is used for increased efficiency and quality of the search. The benefit of distributed web crawler is that it is robust against system crashes and other events, and is adaptable to various crawling algorithms.[7]

4. Parallel Crawler

Different crawlers running in parallel are often called parallel crawler. A parallel crawlers consists of multiple crawling processors called C-procs. The parallel crawlers are affected by page freshness and page visiting.. Parallelization of crawling system is very vital from the point of view of downloading documents in a reasonable amount of time. [7]

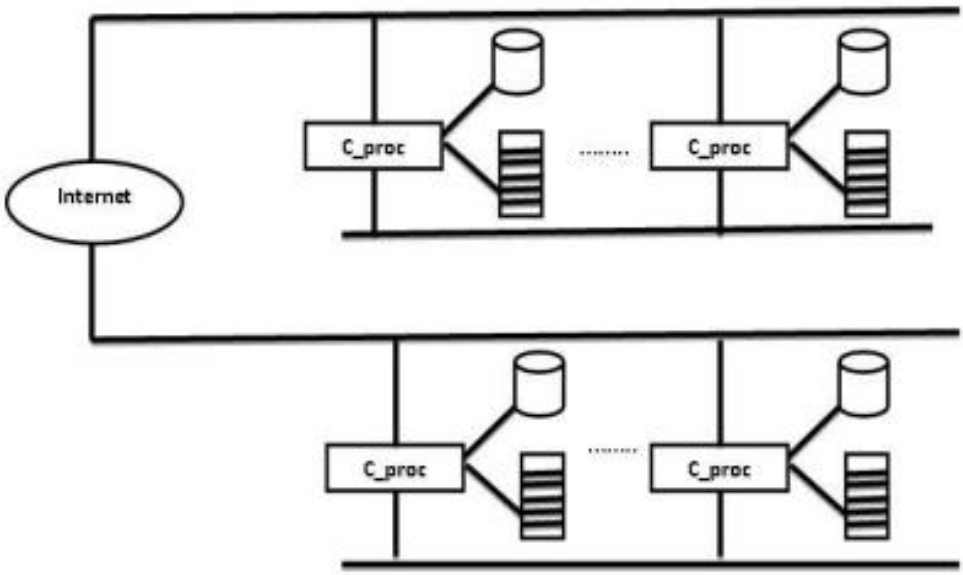


Figure 3.1 Parallel Crawler System Design [9]

We will consider focused crawling in detail. According to [8] there are two main reasons for settling with a focused crawling -

1. The web graph is very mixing as random links lead to random pages with different topics and te pages may get unrelated.

2. The might be a large graph in the pages where the topic coherence persists
The above requirements are very contradictory and this makes using web crawling very useful.

A consequence of the resulting efficiency is that it is feasible to crawl to a greater depth than would otherwise be possible.
Focused crawling is the way to crawl only desired pages.

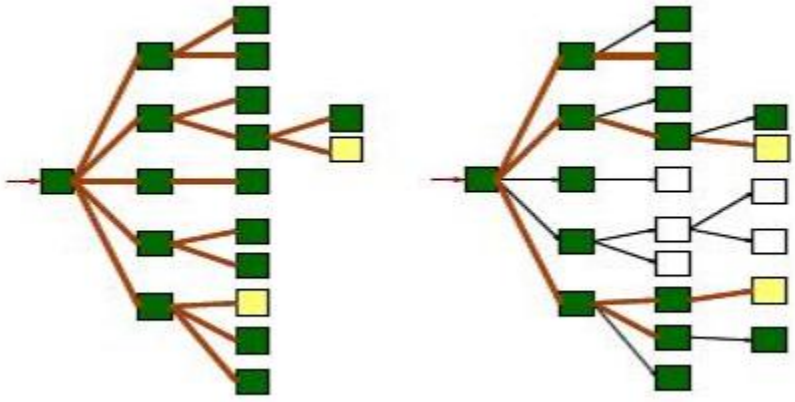


Figure 3.2 (1) General Crawler (2) Focused Crawler

3.4 Content – Based Block Segmentation

Multiple-topic and varying-length of web pages negatively affect the performance of crawlers. Now we see how to segment web pages into parts to simplify the crawling. Different segmentation techniques have different effects on the page improvements.. [10]

1. Fixed-length Page Segmentation - In traditional text retrieval, fixed-length passages, or windows, are used to overcome the difficulty of length normalization. A fixed length passage

contains fixed number of continuous words. For web documents, fixed-length page segmentation is identical to traditional window approach except that all the HTML tags and attributes are removed. Despite its simplicity, fixed-length segmentation is very robust and effective for improving performance, particularly for collections with long or mixed-length documents. The main shortcoming of the fixed-length method is that no semantic information is taken into account in the segmentation process.

2. DOM-based Page Segmentation - DOM provides each web page with a fine-grained structure, which illustrates not only the content but also the presentation of the page. In general, similar to discourse passages, the blocks produced by DOM-based methods tend to partition pages based on their pre-defined syntactic structure, i.e., the HTML tags. There are some approaches that take into account the problem of page segmentation, but there is no consistent way to do it and, to the best of our knowledge, few works are done on applying DOM based page segmentation methods on web information retrieval. The reasons may lie in the following three aspects. First, DOM is still a linear structure, so visually adjacent blocks may be far from each other in the structure and departed wrongly. Secondly, tags such as <TABLE> and <P> are used not only for content presentation but also for layout structuring. It is therefore difficult to obtain the appropriate segmentation granularity. Thirdly, in many cases DOM prefers more on presentation to content and therefore not accurate enough to discriminate different semantic blocks in a web page.

3. Vision-based Page Segmentation - People view a web page through a web browser and get a 2-D presentation which provides many visual cues to help distinguish different parts of the page, such as lines, blanks, images, colors, etc.

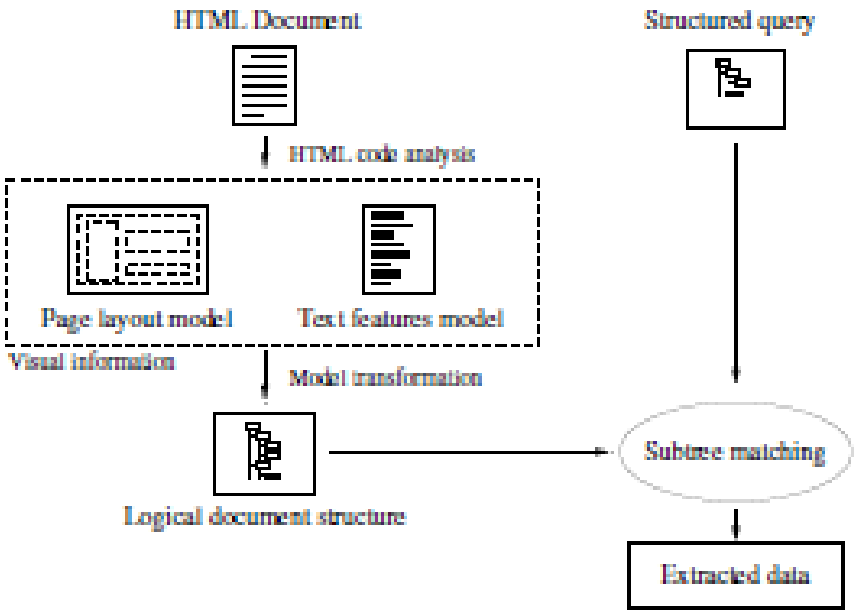


Figure 3.3 A structure of Visual Based Segmentation

Similar to semantic passages, the blocks obtained by VIPS are based on the semantic structure of web pages. Traditional semantic passages are obtained based on content analysis which is very slow, difficult and inaccurate. VIPS discards content analysis and produce blocks based on the visual cues of web pages. This method simulates viewers understanding of web layout structure based on its visual perception. Since the method is totally top-down and the permitted degree of coherence is be pre-defined, the whole page segmentation procedure is efficient, flexible and more accurate from semantic perspective.

Page segmentation can significantly improve the retrieval performance of the crawler. By integrating semantic and fixed-length properties, we could deal with both problems and achieved the best performance. We believe such a block-level analysis of web pages will have the opportunity to significantly enhance the performance of existing commercial search engines.[10] The [4] particularly focuses on the **block based segmentation**. It divides the web page into three major areas –

1. Head – The head part of the page contains the information about the page. In particular it contains the meta information about the page. This information is stored in the meta tag and tells us crucial information about the information contained in the page.

2. Hyperlink – The hyperlinks contain a information about the page, The anchor text is very useful in knowing what the link points to. This information tells about the amount of information on the given topic and subject in it also. The tags can be used to estimate the focus of the linked page in the topic.

3. Body – The body is the part where all the text lies. This part contains the information. This part can used to read information and access the importance of the topic to the page.

The [11] introduce their technique for traversing the DOM tree and selecting relevant content. They target so called data records which are pieces of a web page which are repeating themselves, but with a different content. Their algorithm is divided in several steps. In the first step it gets the content candidates node using a BFS – like algorithm. Data records are defined as tags on the same level of DOM tree, containing repetitive children sequences and having similar parent. The parent is denoted as data region. In case no nodes on a particular level of BFS satisfy the definition, the next tree level is inspected. Output of the first stage is a list of all data regions identified on the page. Other stages only filter results the algorithm gained in the first stage. Following observations are used for filtering heuristics: 1. Relative to the whole page, data records (and subsequently the whole data region) have large size 2. Data Records are usually repeated more than three times on a page 3. A regular expression can be devised for description of data record. Since all data records share the same template, it will apply on all of them 4. Data Records usually consist of a small amount of HTML tags. After the list of data regions is filtered, every data region has to be assigned its relevancy score. The sorting function described in determines the size of area taken by data records by counting characters and images each data record contain. Elements representing free space are taken into account as well. Data region with the best score is considered to be the main content of the page. [11]

3.5 INDEGREE

The INDEGREE algorithm is explained in [12]. A simple heuristic that can be viewed as the predecessor to all Link Analysis Ranking algorithms is to rank the pages according to their. The popularity of a page is measured by the number of pages that link to this page. We refer to this algorithm as the INDEGREE algorithm, since it ranks pages according to their in-degree in the graph G. That is, for every node i, $a_i = |B(i)|$. This simple heuristic was applied by several search engines in the early days of Web search. INDEGREE algorithm is not sophisticated enough to capture the authoritativeness of a node, even when restricted to a query dependent subset of the Web.

3.6 PAGERANK

PageRank ranks pages based on the web structure. PageRank figures that when one page links to another page, it is effectively casting a vote for the other page. The more votes that are cast for a page, the more important the page must be. Also, the importance of the page that is casting the vote determines how important the vote itself is. PageRank calculates a page's importance from the votes cast for it. [13]

PageRank [1] is defined as follows: We assume page A has pages TI...Tn which point to it. The parameter d is damping factor which can he .set between 0 and 1. We usually set d to 0.85. Also C(A) is defined as the number of links going out of page A. The PageRank of u page A is given as follows:

PR(A) = (I -d) PR(TI) +d(p PR(Tn) C(Tl) +...+- C(Tn)).

Note that the PageRanks form a probability distribution over Web pages, so the sum of all Web pages' PageRanks will be one. PageRank or PR(A) can be calculated using a simple iterative algorithm, and corresponds to the principal eigenvector of the normalized link matrix of the Web.

The Page Rank algorithm is given by

1) Calculate page ranks of all pages by following formula:

PR(A) = (1-d) + d (PR(T1)/C(T1) + ... + PR(Tn)/C(Tn))

Where

PR(A) is the PageRank of page A,

PR(Ti) is the PageRank of pages Ti which link to page A,

C(Ti) is the number of outbound links on page Ti and

d is a damping factor which can be set between 0 and 1, but it is usually set to 0.85

2) Repeat step 1 until values of two consecutive iterations match.

The features of PageRank algorithm are

1. It is the query independent algorithm that assigns a value to every document independent of query.
2. It is Content independent Algorithm.
3. It concerns with static quality of a web page.
4. Page Rank value can be computed offline using only web graph.
5. Page Rank is based upon the linking structure of the whole web page.
6. Rank does not rank website as a whole but it is determined for each page individually.
7. Page Rank of pages Ti which link to page A does not influence the rank of page A uniformly.
8. More the outbound links on a page T, less will page A benefit from a link to it.
9. Page Rank is a model of user's behavior
10. It is easily linked to mathematical background.[13]

The disadvantages of PageRank are given below as :

1. It favours the older pages.

2. It has the problem of link farm.

3. It has the problem that some sites may buy links.

The problem can be restated as a linear system.

$$G = \pi S + (1 - \alpha)/v.$$

Transforming $\pi G = \pi$ to $0 = \pi - \pi G$ gives:

$$\begin{aligned} 0 &= \pi - \pi G \\ &= \pi I - \pi (\alpha S + (1 - \alpha)/v) \\ &= \pi (I - \alpha S) - (1 - \alpha)(\pi/) v \\ &= \pi(I - \alpha S) - (1 - \alpha) v \end{aligned}$$

The last equality follows from the fact that π is a probability distribution vector, so the elements of π are nonnegative and sum to 1.

In other words, $\pi = 1$. Thus,

$$\pi (I - \alpha S) = (1 - \alpha) v,$$

which means π solves a linear system with coefficient matrix $I - \alpha S$. [19]

Decreasing Number of Iterations using Normalized optimized technique

In [13] it has been suggested that the number of iteration of the PageRank algorithm can be decreased by using the average value of page rank of every page and dividing the value of each page by that value.

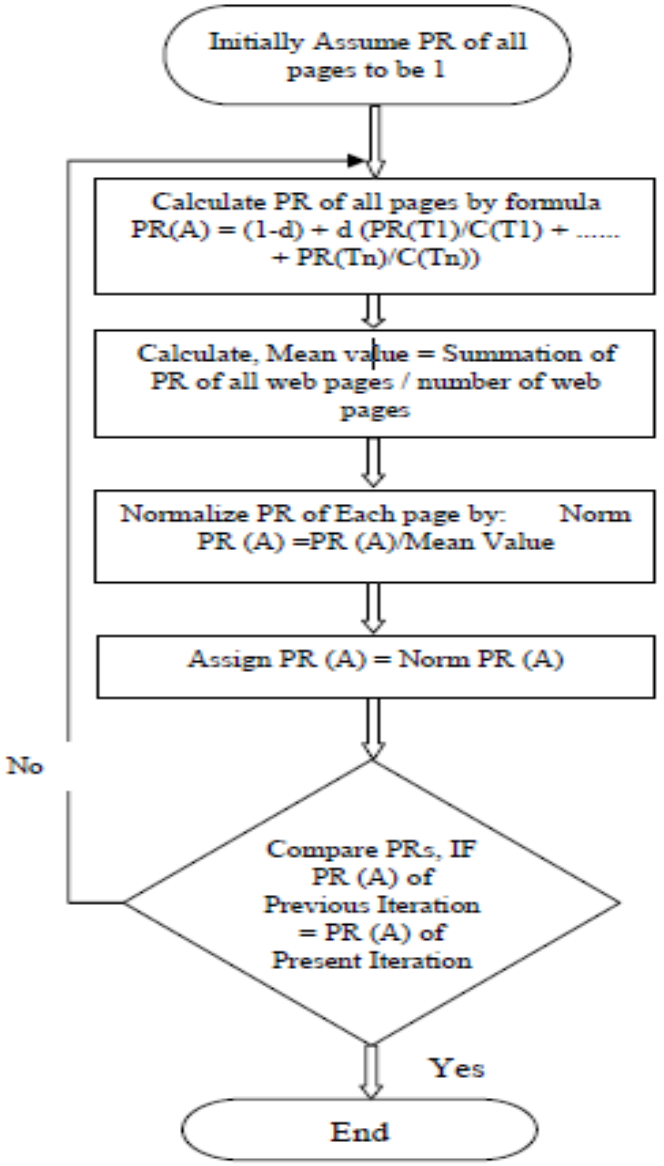


Figure 3.4 Normalized Optimized PageRank Algorithm

The algorithm is shown below

1) Initially assume PAGE RANK of all web pages to be any value, let it be1.

2) Calculate page ranks of all pages by following formula

PR(A) = .15 + .85 (PR(T1)/C(T1) + PR(T2)/C(T2) +……. + PR(Tn)/C(Tn))

Where

T1 through Tn are pages providing incoming links to

Page A

PR(T1) is the Page Rank of T1

PR(Tn) is the Page Rank of Tn

C(Tn) is total number of outgoing links on Tn

3) Calculate mean value of all page ranks by following formula :-

Summation of page ranks of all web pages / number of web pages

4) Then normalize page rank of each page

Norm PR (A) = PR (A) / mean value

Where norm PR (A) is Normalized Page Rank of page A and PR (A) is page rank of page A

5) Assign

PR(A)= Norm PR (A)

6) Repeat step 2 to step 4 until page rank values of two consecutive iterations are same. The pages which have the highest page rank are more significant pages.

Page quality, as represented by PageRank, in the context of company home pages and in certain search engine optimizer webs, would be just as useful if based on indegree. This finding, in combination with previous PageRank failures, casts serious doubt on the usefulness of PageRank over indegree. [18]

3.7 RatioRank

The algorithm takes the weights of the inlinks and outlinks based on the popularity of the links in a defined ratio. As the popularity meant for number of inlinks and outlinks to that link of the page. The algorithm also considers the number of times the user visits the inlink of any webpages.

The algorithm is described as

1. Take the link structure of the retrieved webpages from the crawler.

2. Obtain the web graph from the link structure of the retrieved webpages.

3. Assign 1 as the initial ranking to all the webpages.

4. Calculate the weights of inlinks and outlinks using equation

$$W^{in}_{(v,u)} = I_u / \sum_{p=R(v)} I_p$$

$$W^{out}_{(v,u)} = O_u / \sum_{p=R(v)} O_p$$

5. Apply the RatioRank as in the equation

$$RR(u) = (1-d) + d \sum_{v \in B(u)} \frac{(V_u * .7 * W^{in}_{(v,u)} + .3 * W^{out}_{(v,u)}) RR(v)}{TL(v)}$$

6. Repeat the process iteratively until ranks of all webpages are stable means same in two consecutive iteration.

The algorithm gives better results in terms of the relevancy of page as the inlinks, outlinks and number of visits are also considered in the algorithm and limitation of random surfer that it stops on reaching a page with no path outside is overcomed here by using the outlinks to calculate the score of the page. With the limitation of the ranking of any webpage may be intentionally increased, in RatioRank, a specialized crawler is needed which must be capable of counting the visit count, with the use of three features, it increases the computational complexity of the process.

The [14] RatioRank reduces the number of iteration to reach the normalized ranks to the pages.

3.8 HITS

HITS(Hyper – text induced topic search) algorithm ranks the web page by processing in links and out links of the web pages. In this algorithm a web page is named as authority if the web has many inlinks and a web page is named as HUB if the page has many of outlinks.

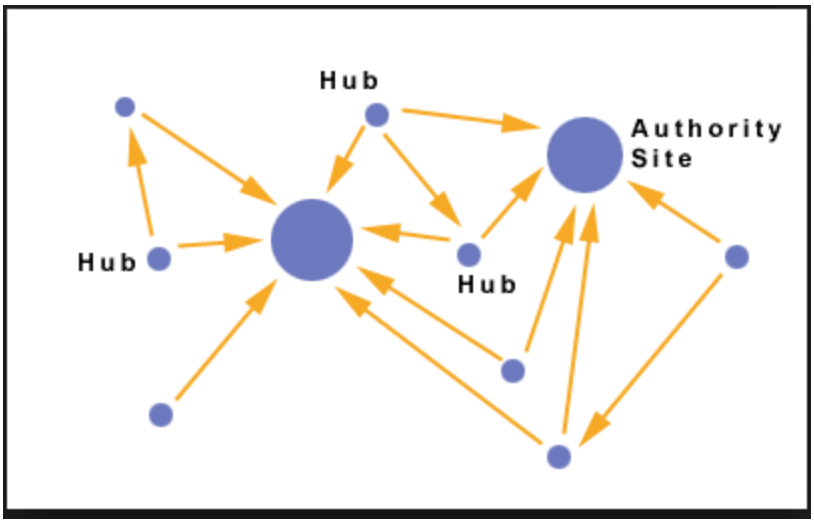


Figure 3.5 Hubs and Authorities

HITS is technically, a link based algorithm. In HITS algorithm, ranking of the web page is decided by analyzing their textual contents against a given query. After the pages are collected then the ranking is done by the hyperlink structure only and the contents of the page hence are neglected.

Original HITS algorithm has some problems which are given below.

(i) The popular sites are given high rank value even if they are not relevant to the given query for the algorithm.

(ii) The search moves away from the topic if the hub has multiple topics as equal rank is given to each link.

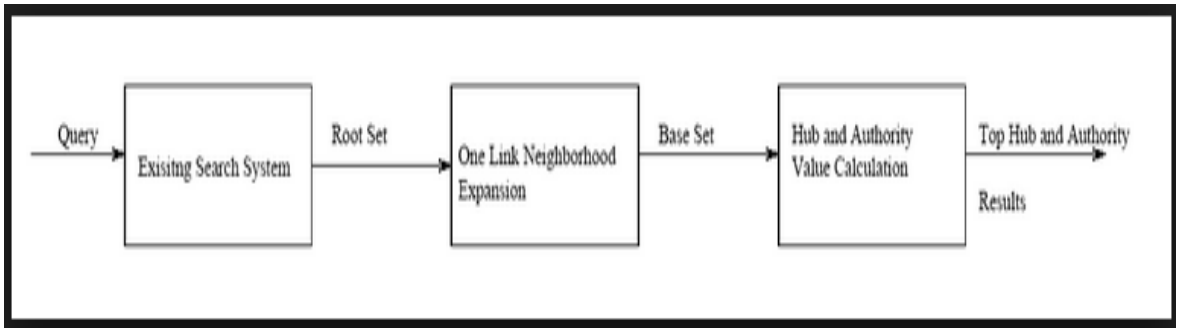


Figure 3.6 HITS

To minimize the problem of the original HITS algorithm, a clever algorithm is proposed. Clever algorithm is the modification of standard original HITS algorithm. This algorithm provides weight to link according to link query and the destination of link it has. The text of the link is used to determine the weight of the page on topic and a large hub is broken into smaller pages. This is used to make the information specific to the page.[15]

Another limitation of standard HITS algorithm is that it assumes equal weights to all the links pointing to a webpage and it fails to identify the facts that some links may be more important than the other. To resolve this problem, a probabilistic analogue of the HITS (PHITS) algorithm is there. It is able to identify authoritative document as claimed by the author. PHITS gives better results as compared to original HITS algorithm. The PHITS can tell the probabilities of authorities for comparison whereas the HITS only provides the single magnitude. [15]

The algorithm for HITS is below

Input: Base set S, Output: A set of hubs and a set of authorities.

1. Let a page p have a authority weight x_p and hub weight y_p. Pages with relatively large weights x_p will be classified to be the authorities, similarly hubs with large weights y_p

2. The weights are normalized to have the squares total of each type be 1.

3. For a page p, the value of x_p is now modified to weight y_p of all pages q pointing to p.

4. The sum of of each q page x_p is now the y_p of page.

5. Continue with step 2 unless a termination condition true.

6. Output the set of pages with the largest x_p weights i.e. authorities, and those with the largest y_p weights i.e. hubs. [2]

3.9 Semantics based crawling

Semantic based crawling is which is based both on the semantic content of the URL and all its parent pages along with the importance metric give a new method for prioritizing the URL queue for crawling by considering both the semantic and link structure of the web. [16]

3.10 Semantic Web

The **Semantic Web** is the extension of the World Wide Web that enables people to share *content* beyond the boundaries of applications and websites. The Resource Description Framework (RDF) is a language that has been developed for defining the web resources and the relation between them. OWL is already being successfully used in many applications. This success brings with it, however, many challenges for the future development of both the OWL language and OWL tool support. [20]

3.11 Comparison of ranking algorithms

Web Mining is used to extract the useful information from very large amount of Web data. The usual search engines usually result in a large number of pages in response to user's queries. while the user always wants to get the best in a short span of time so it does not bother to navigate through all the pages to get the required ones. The page ranking algorithms, which are an application of web mining, play a major role in making the user search navigation easier in the results of a search engine. The PageRank and ratiorank algorithm give importance to links rather than the content of the pages, the HITS algorithm stresses on the content of the web pages as well as links, while the Indegree algorithm only focuses on the number of inlinks to the page. Depending upon the technique used, the ranking algorithms give a different order to the resultant pages. The PageRank and ratiorank return the important pages on the top of the result list while others return the relevant ones on the top. A typical search engine may deploy a particular ranking algorithm depending upon situation. As a guidance, the algorithms which equally consider the relevantly as well as importance of a page which should be developed so that the quality of search results can be improved. [2]

4. Problem Statement and Proposed Model

Now we present our work and the problem it addresses. Firstly we introduce the problem and then we present our model which addresses the problem. Today the searches in internet have not the kind of accuracy and totality as desired by the searcher. The irrelevant or unrequired results are very unsatisfactory to the user. Also from the point of the developers the process of ranking is a very long and time consuming process. Therefore we present framework for search engine which increases the performance of search engine and also makes the results more relevant and fast. The problem is stated as improving the efficiency and quality of search engines.

We propose a model for ranking where the content of results is improved and the the time of search engine is also decreased.

We use the content – based segmentation as discussed in Chapter 3 and section 3.4. But instead of crawling, we use this technique in raking by using the score of content – based segmentation as the initial score for the ranking algorithm. Then we use the ranking algorithm. The ranking algorithm used is the page rank algorithm as explained in Chapter 1 and section 1.5 and Chapter 3 and section 3.6. But we modify the algorithm to improve the efficiency and performance of the algorithm. The changes made are that we do not equally give rank rather we distribute the rank according to the proportion of tf-idf. In Chapter 3 and section 3.6 under the heading "**Decreasing Number of Iterations using Normalized optimized technique**" we used the normalized optimized technique to decrease the number of iterations. We modify this technique to decrease the number of iterations even further. Secondly we use the differential normalized optimization technique to proceed the algorithm. This technique decreases the number of iteration required by the algorithm.

Next we explain the model in detail and also reasoning the modification we made in the existing system. This would provide the insight into the working and calling of the system.

4.1 Seed Selection

The process of seed selection is key issue in the performance of search engine. It determines the overall quality of the search results. This is the most important part of the information collection. Therefore, good seeds would be linked to good seeds and give the proper results. The process of seed selection in the system is done by querying the search engine google for the term and getting the top ten or twenty results and used them as seed.

4.2 Content – Based Block Segmentation of crawled pages

In this algorithm we use the Content – Based Block Segmentation block segmentation of the in pages to calculate score of the pages. This value of score is used to initialize the initial score of the pages. More specifically the value of the score will be used as the rank of the page in the first step of the ranking algorithm. This is a change in the original algorithm which uses the value to initialize all the pages. This value is calculated during the crawling process for each page and stored in memory. The content based segmentation is used to calculate the number of occurrences of the search term in the different part of the page. The parts were divided on the basis of HTML tags they were in. The parts considered by the algorithm are

1. <HEAD>……..</HEAD> - The contents of the <head> tag were considered first. This was considered to be the most important position for the search term to be in the whole page. The maximum sore was given to each presence of the search term in the part of the page. The reason for this was that head contains information that pertains to the information sources and areas of the information contained in the page and external files. Also the tag contains the <meta> tag inside it. <meta> tag provides the information contained in the page.This provide information about the information in the page called the meta information. Therefore, if the page contains term in its head tag, then the page definitely provides some information about the term. The head part is the part lying between the <head>…….</head> part. The part is given the maximum score. Different values of the score have been experimented with.

The part was extracted using the regular expression

"<head[^>]*>(.*?)</head>"

The frequency of the term in the string was seen and then multiplied by the value of term in presence of head. This gave the head score of the Page. This score was added to the overall score of the page to give it the total score of the page.

2. <A>…….</A> - Then we considered the contents of the <A>…..</A> tag and gave the second largest score. The second largest score was given to each presence of the term in the anchor text of a link. The reason is the anchor text defines the contents of then page the link directs to. Therefore, the presence of term in anchor text provides an indication on the contents of the link and thus must add to the score of the current page because it provides a link to a page containing the term.

The anchor text was obtained by the regular expression

<a[^>]*>(.*?)

The frequency of the term in the string was seen and then multiplied by the value of term in presence of <A></A>. This gave the link anchor score of the Page. This link anchor score was added to the overall score of the page to give it the total score of the page.

3. <body>……</body> - The last part considered was the <body > of the of the HTML page and given the least score. The presence of the term in body indicates that the term is present in the page that indicates that the information about the term is present In the page. Therefore the score was added to the score of the page. This aprt was given the minimum score because the page only contains the information and therefore there should be high frequency of term so that the information could be considered relevant.

The expression used to get the contents of the body tag is

body score = body number * (count number of occurrences in body – link anchor occurrences – head occurrences)

The process can be explained as by the architecture

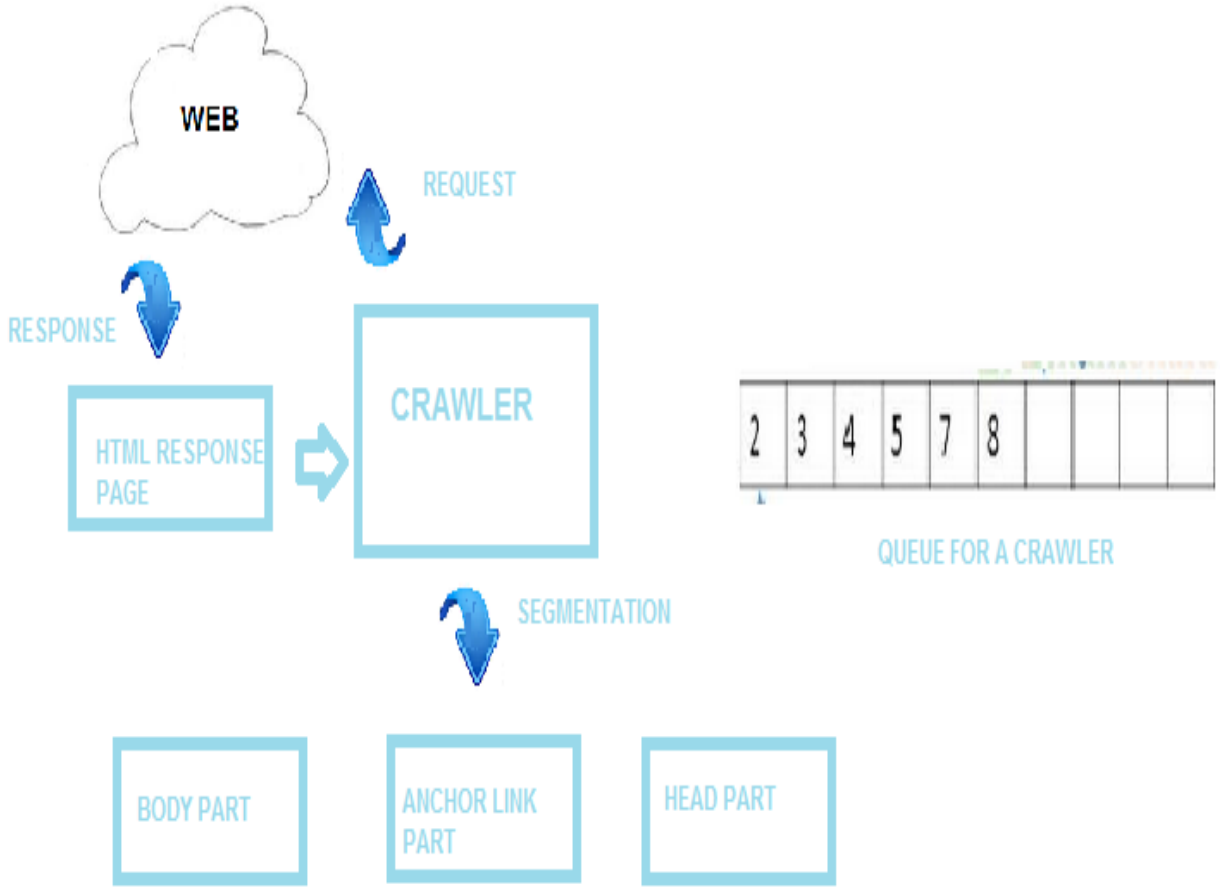


Figure 4.1 Content – Based Segmentation of page

The drawn shows that the segmentation breaks the page into its different contents. The next step is ti get the score of each part. Now we show the extraction of each link from the page.

4.3 Extracting the links

Now after calculating and storing the initial rank of each page the next step is to extract the links from the page and add them to the queue. The queue is maintained by the crawler. The next URL to be crawled is taken from the queue and its initial rank is calculated and the links are extracted.

This helps to crawl the required number of pages from the small number of seed URLs. The process of link extraction is shown below

1. Get the next URLs from the queue

2. Get the contents of the page and get the score using the formula in chapter 4 and section 4.2

3. Store the contents of the string.

4. Get all the links in the page using the regular expression

<a\\s+href\\s*=\\s*\"?(.*?)[\"|>]

5. Add the hyperlinks to the queue of the URLs

However in the system there are some issues to be addressed.

1. Duplicate URLs – There might be multiple pages pointing to the same URL. There there might be duplicate URLs in the queue. This is solved using the hashmap to store all the crawled URLs. So when we get a URL we check if it is not already crawled and crawl it if it is not already crawled.

2. Secondary Links – All pages will be having links which are not hyperlinks and also malformed links. These links are secondary links. We ignore the secondary links by checking with the regular expression and only to allow well – formed hyperlinks. We eliminate the mail links, the javascript links and the anchor links, This filtering helps us to get only the required and valid links. This is very useful to do here. This helps the search engine to only get the contents of the required links.

The drawing given below explains the process of link extraction and the storing of link.

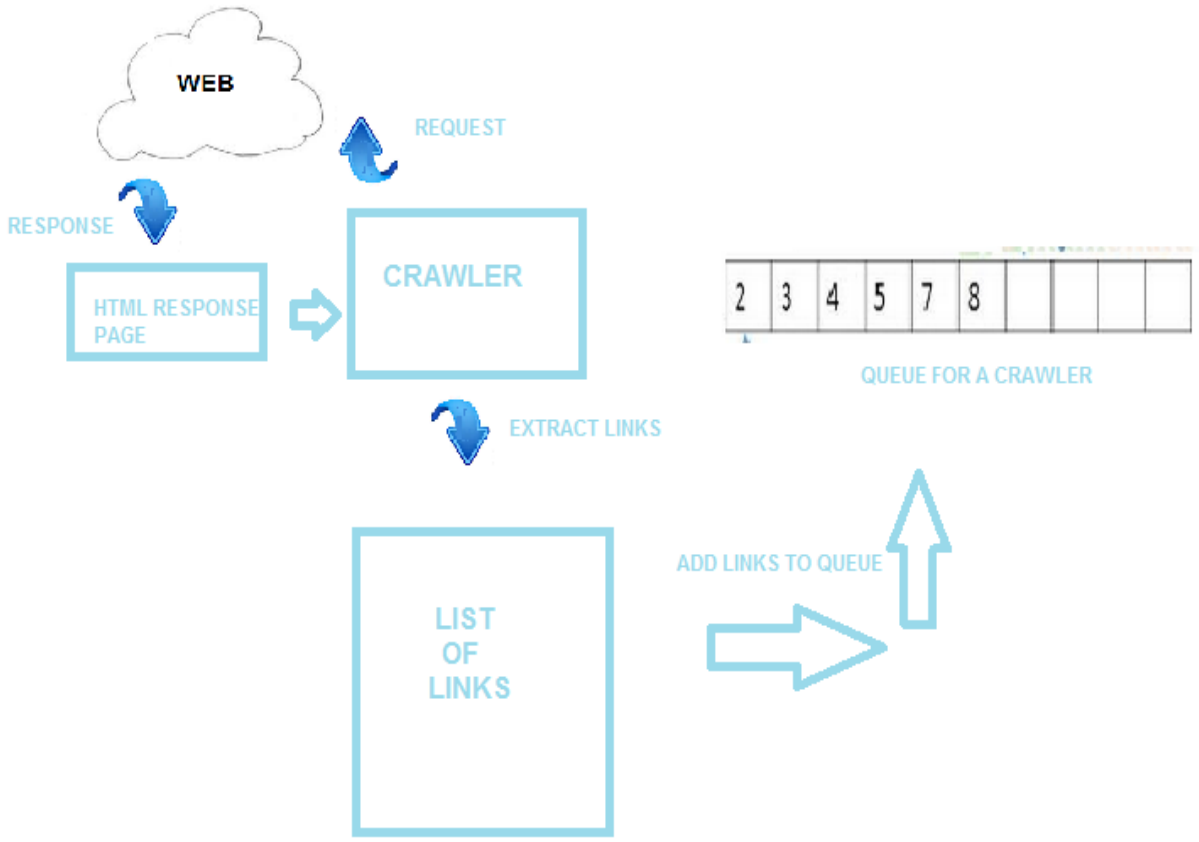


Figure 4.2 Extraction of the links from page

Now after extracting all the links and crawling them we finally have our following map telling the URLs and their respective scores.

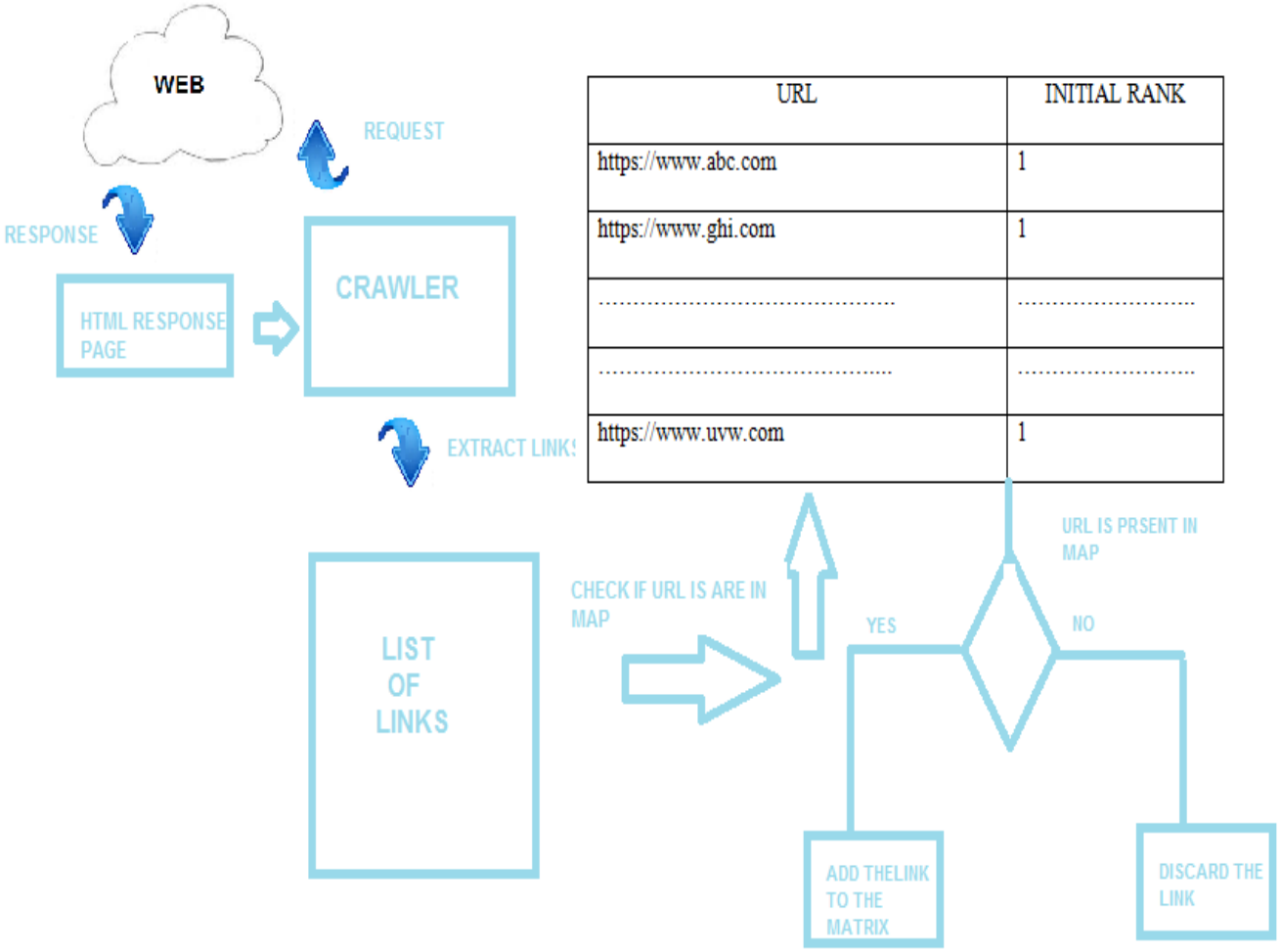
URL	INITIAL RANK
https://www.abc.com	1
https://www.ghi.com	1
……	……..
…….....	……..
https://www.uvw.com	1

Table 4.1 Table containing the Hyperlinks and their Initial rank

4.4 Forming the Matrix of pages

Now after crawling all the pages we now start the process of ranking. For this we require a matrix to indicate the hyperlink structure of the graph of linked pages. This representation is obtained by the help the map we created in the previous step. The step are discussed below

1. For every page in map
    a. Get the contents of the map
    b. Scan the contents of the and discover all the links
    c. For each link
        i. Find if the link exists in map
        ii. If yes add the link showing the link from the page to the linked page
        iii. Otherwise ignore the link
2. Get the desired matrix
3. Compress the matrix
4. Done



URL	INITIAL RANK
https://www.abc.com	1
https://www.ghi.com	1
.....	.....
.....	.....
https://www.uvw.com	1

Figure 4.3 Map To Create The Link Matrix

4.5 Ranking Of The Results

Now the task of the ranking algorithm starts. The algorithm will use the modified algorithm. This will improve the performance of the crawler. This will help to order the results according to their relevance.

The algorithm is essentially PageRank algorithm but with the following differences

1. The value of initial rank of each page is the score calculated during the crawling process.
2. Instead of using the value of probabilities it gives the full score of the links.

3. Instead of using the score value directly, we divide it by the product of difference of previous minimum and maximum score and average score of this iteration.

4. Instead of absolute matching between iterations we have set an error limit of $1*10^{-9}$.

The advantage of these changes is that by using the initial value as score value as score the more important pages are getting better score. The links of important pages get more score. The division of score allows the better rate of execution of the algorithm. Allowing, the error lets the unnecessary iterations to be ignored.

Now we discuss the algorithm for the system –

Rank

Input matrixrank[j][j] containing the links of the graph

mapscore[j][j] containing score of each page

number which is the number of URLs

tempscore[j] containing the temp score of each and every page initialized to 0.0.

outlinks[i] contains the number of outlinks of page to distribute score among all outlinks.

1. proceed=false

2. while proceed=false

i. for i=1 to number

a. total=0

b. for k=1 to number

A. if matrixrank[i][k]=1

temp[i]=temp[i]+score[k]/outlinks[k]

ii. difference = (max – min)/2

iii. max = maximum value of score

iv. min = minimum value of score

v. for k=1 to number

a. tempscore[k]=tempscore[k]/(differerence*(max-min)*total/number)

vi. proceed=true

vii. for i=1 to number

a. if tempscore[i]-mapscore[i]>0.000001

A. proceed=false

B. break

viii. for i=1 to number

a. mapscore[i]=tempscore[i]

ix. for i=1 to number

a. tempscore[i]=0.0

3. Results are obtained

4. End

The result of the algorithm is the score of every URL. This is the final score. The score is then sorted. Then the result is displayed according to the ranking algorithm ranks.

The next we show the working of the ranking algorithm.

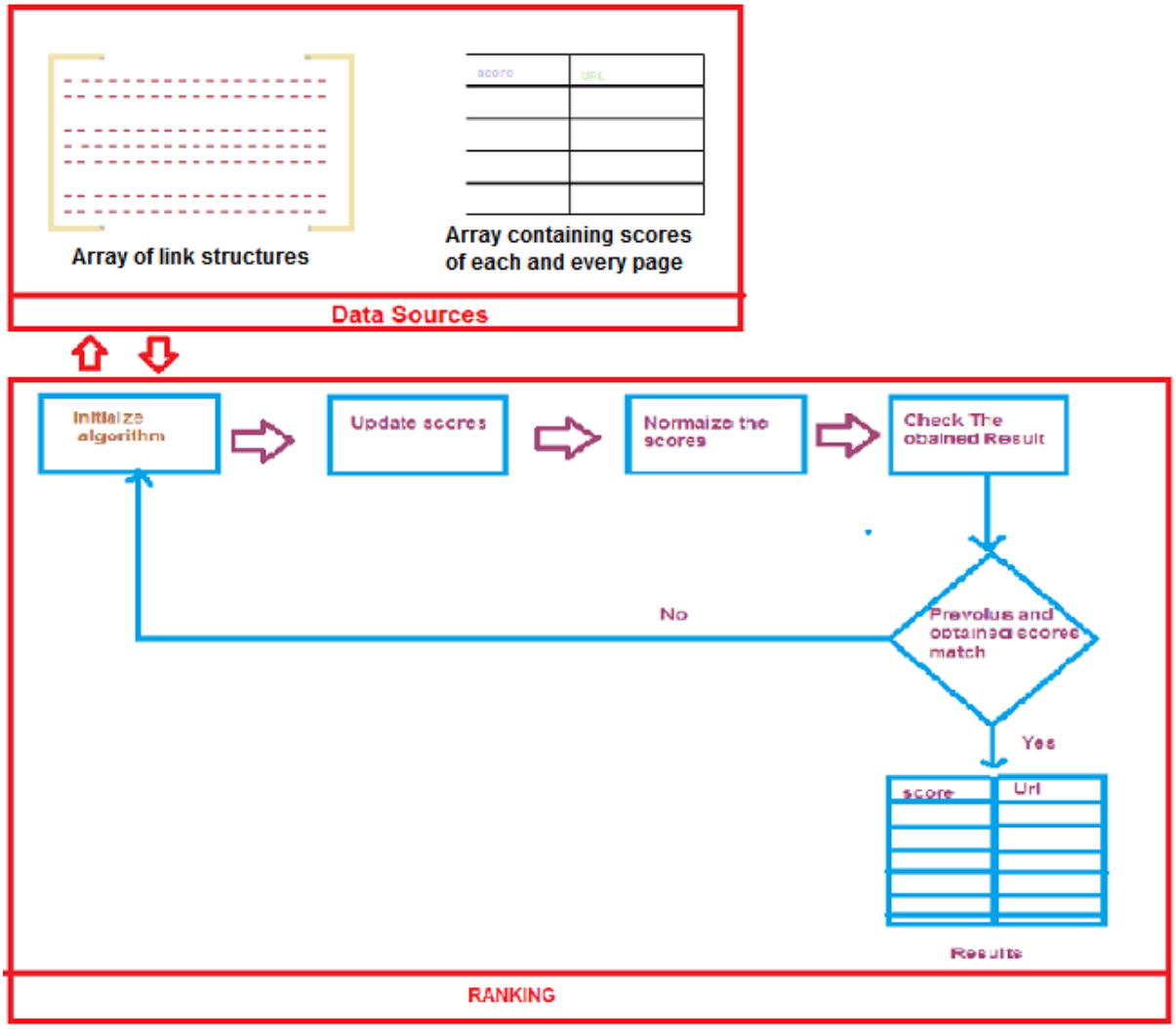


Figure 4.4 Ranking algorithm for the crawler

The ranking give the desired results. Now the search engine is ready to give answers to the user query. The results can now be given to the user. The order determines the position of the results in the result of the query.

5. Results and analysis

This section describes the implementation of the system. The proposed system was implemented and the results were taken. The implementation was results oriented. The calculated values gave the indication of the algorithm.

5.1 Experimental Setup

The proposed system was implemented to check for its performance in java. Our algorithm, "Score-based initialized, proportionally distributed rank and differentiated-normalization functioning" was compared against the standard algorithms PageRank and Normalized PageRank. The comparison was done with varying number of nodes and value assigned to the occurrence in head, link and body. The parameters were used with different algorithms.

5.2 Issues and resolution of issues

The different issues that arised during the development were

Issue 1 The some pages had no outgoing or all out going links had score 0 link and this made the total zero which lead to divide by zero exception.

Resolution 1 In the initial score of every page 1 was added to prevent.

Issue 2 The range of numbers went out of the range of fractional part of the storage leading to NaN.

Resolution 2 The class Bigdecimal was used to store the scores.

5.3 Input

The input and the quantities to be tested were as follows

1. Number of pages to be ranked

2. The value of occurrence in head, link and text.

3. The query to be used.

4. Algorithm to be used

5.4 Data structures used in the implementation

1. map_scsore – hashmap containing two fields one a integer and one a Bigdecimal. It is used to map the page number to their scores.

2. map_page_number_to_url – hashmap containing two fields one a String and one a integer. It is used to map page number to page URL.

3. map_url_to_page_number – hashmap containing two fields one String and one a integer. It is used to map the page URL to page number.

4. matrix – two – dimensional matrix of integers. It is used to contain the link structure of the discovered pages.

5. list –list of URL that are discovered by links from the previous pages. It is used to store the pages to be crawled.

5.5 Studying the effects of different values for occurrences in head, link and body

Now we study the effects of choosing different values of the occurrences in head, link and text. The values were applied on the proposed algorithm and the results were as follows

Search term Operating system

1. text=1, link=2, head=4

The results obtained were

http://en.wikipedia.org/wiki/Operating_system
http://electronics.howstuffworks.com/tech
http://www.howstuffworks.com/
http://electronics.howstuffworks.com/
http://computer.howstuffworks.com/
http://computer.howstuffworks.com/operating-system.htm
http://www.computerhope.com/os.htm
http://www.webopedia.com/TERM/O/operating_system.html
http://www.sigops.org/osr.html
http://whatsmyos.com/

Table 5.1 Results with text=1, link=2, head=4

2. text=1, link=2, head=3

The results obtained were

http://en.wikipedia.org/wiki/Operating_system
http://electronics.howstuffworks.com/tech
http://www.howstuffworks.com/
http://electronics.howstuffworks.com/
http://computer.howstuffworks.com/
http://www.computerhope.com/os.htm
http://computer.howstuffworks.com/operating-system.htm
http://www.webopedia.com/TERM/O/operating_system.html
http://www.sigops.org/osr.html
http://whatsmyos.com/

Table 5.2 Results with text=1, link=2, head=3

3. text=1, link=3, head=4

The results obtained were

http://en.wikipedia.org/wiki/Operating_system
http://sosp.org/
http://www.computerhope.com/os.htm
http://computer.howstuffworks.com/operating-system.htm
http://courses.cs.vt.edu/csonline/OS/Lessons/
http://www.sigops.org/
http://windows.microsoft.com/
http://whatsmyos.com/
http://www.computerhope.com/
http://www.computerhope.com/oh.htm

Table 5.3 Results with text=1, link=3, head=4

4. text=1, link=1, head=4

The results obtained were

http://en.wikipedia.org/wiki/Operating_system
http://www.computerhope.com/oh.htm
http://www.computerhope.com/tips/
http://www.computerhope.com/jargon.htm
http://www.computerhope.com/
http://www.computerhope.com/os.htm

http://computer.howstuffworks.com/operating-system.htm
http://www.webopedia.com/TERM/O/operating_system.html
http://www.sigops.org/osr.html
http://www.computerhope.com/

Table 5.4 Results with text=1, link=1, head=4

5. text=1, link=1, head=3

The results obtained were

http://en.wikipedia.org/wiki/Operating_system
http://www.computerhope.com/oh.htm
http://www.computerhope.com/tips/
http://www.computerhope.com/jargon.htm
http://www.computerhope.com/
http://www.computerhope.com/os.htm
http://computer.howstuffworks.com/operating-system.htm
http://www.sigops.org/osr.html
http://whatsmyos.com/
http://www.webopedia.com/TERM/O/operating_system.html

Table 5.5 Results with text=1, link=1, head=3

The tests suggest that the results obtained by different value of text, link and head affect the accuracy of search engine in a very small way. Therefore, the values are not important as far as the accuracy is of concern, but the time required to complete the iterations is low if the value are closer. This means the values 1,3,4 are suitable values and we will continue with these values.

5.6 Comparison Of The Algorithms

The comparison of the algorithm will be done by supplying the same input to algorithm and seeing the results. The time and performance of algorithms will be compared. The results are then compared. The algorithms to be considered are

1. Proposed algorithm

2. PageRank

The test are

Output number of iterations

1. search string integer

   i. pages to be crawled 10

   ii. pages to be crawled 50

   iii. pages to be crawled 100

2. search string process

   i. pages to be crawled 10

   ii. pages to be crawled 50

   iii. pages to be crawled 100

3. search string people

   i. pages to be crawled 10

   ii. pages to be crawled 50

   iii. pages to be crawled 100

4. search string english

i. pages to be crawled 10

ii. pages to be crawled 50

iii. pages to be crawled 100

5. search string sports

i. pages to be crawled 10

ii. pages to be crawled 50

iii. pages to be crawled 100

search string integer

Number of iterations ↓	Proposed algorithm	PageRank
10	12	369
30	13	105
50	13	76

Table 5.6 Results with search string integer

search string process

Number of iterations ↓	Proposed algorithm	PageRank
10	15	43
30	21	37
50	22	54

Figure 5.7 Results with search string process

search string people

Number of iterations ↓	Proposed algorithm	PageRank
10	10	23
30	12	37
50	13	98

Table 5.8 Results with search string people

search string english

Number of iterations ↓	Proposed algorithm	PageRank
10	17	45
30	25	35
50	30	34

Table 5.9 Results with search string english

search string sports

Number of iterations ↓	Proposed algorithm	PageRank
10	13	127
30	22	204
50	28	131

Table 5.10 Results with search string sports

The results show that for large number of nodes the system yields better results. It also shows that the number of cycles do not increase with the number input pages as per the order. The

number of cycles increase with diameter of the graph. The number of cycles depends on the link structure of the pages. More the levels in the graph more the time will be taken to and hence the number of loops are more.

The main advantage the system has that it can reduce the number of cycles in the main loop. This provides the opportunity for parallelism.

However due large precision in floating point required by the system it not suitable to large scale system. But it is very useful in peer to peer networks or domain based ranking. However with better computing resources it can be applicable in large scale system as well also.

6. Conclusion and Future Work

The problems in the current search engines were studied and analyzed in the above work. The existing systems were studied and the negative points were seen. In particular, the lack of consideration of contents and importance of the page was seen. The model was then proposed and implemented. The model used content – based segmentation for content analysis and proportional initialization of ranking among the pages. The differential normalized technique was used to make the system more efficient and to make it more affective.

The results show that the system effectively produces good results and also decreases the amount of time required by the system. But due to lack of efficient handling of floating numbers, the system run slowed down. But the system has degree of parallelism which makes it very fast in the parallel computing. This is a big advantage with the modern computing environment. However, due to the lack of parallelism in the experiment, the time taken for crawling and ranking was still huge. Also the process of seed selection leads gives us external seed. So, there must be more secure way of seed selection. The model will perform better in the desired conditions.

In future the work will be concentrated on improving the process of seed selection and using more secure technique of seed selection. Also there will be focus on making the index and crawl less on query. This would make the query retrieval much quick. Further, the model will be using strategies like error detection and correction, fast links, synonym detect, and query weighting. This would lead to improved performance of crawler.

REFERENCES

[1] Sergey Brin and Lawrence Page," The anatomy of a large-scale hypertextual Web search engine, Computer Networks and ISDN Systems 30 (1998) 107- 117.

[2] Neelam Duhan, A. K. Sharma and Komal Kumar Bhatia, "Page Ranking Algorithms: A Survey", 2009 IEEE International Advance Computing Conference (IACC 2009).

[3] Juan Ramos, "Using TF-IDF to Determine Word Relevance in Document Queries", Department of Computer Science, Rutgers University, 23515 BPO Way, Piscataway, NJ, 08855.

[4] Bireshwar Ganguly and Devashri Raich, "Performance Optimization of Focused Web Crawling Using Content Block Segmentation", 2014 International Conference on Electronic Systems, Signal Processing and Computing Technologies.

[5] Dennis Fetterly, Nick Craswell, and Vishwa Vinay, "Measuring the Search Effectiveness of a Breadth-First Crawl", M. Boughanem et al. (Eds.): ECIR 2009, LNCS 5478, pp. 388–399, 2009. Springer-Verlag Berlin Heidelberg 2009

[6] FILIPPO MENCZER, GAUTAM PANT and PADMINI SRINIVASAN, "Topical Web Crawlers: Evaluating Adaptive Algorithms", ACM Transactions on Internet Technology, Vol. V, No. N, February 2003, Pages 1 - 38.

[7] Trupti V. Udapure, Ravindra D. Kale and Rajesh C. Dharmik, " Study of Web Crawler and its Different Types", IOSR Journal of Computer Engineering (IOSR-JCE) e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 16, Issue 1, Ver. VI (Feb. 2014), PP 01-05.

[8] Soumen Chakrabarti, Martin van den Berg and Byron Domc, "Focused crawling: a new approach to topic-specific Web resource discovery", 1999 Published by Elsevier Science B.V.

[9] Junghoo Cho and Hector Garcia-Molina, "Parallel Crawlers", WWW2002, May 7–11, 2002, Honolulu, Hawaii, USA. ACM 1-58113-449-5/02/0005

[10] Deng Cai1, Shipeng Yu, Ji-Rong Wen and Wei-Ying Ma, "Block-based Web Search", SIGIR'04, July 25–29, 2004, Sheffield, South Yorkshire, UK.

[11] Jan Zelený, "WEB PAGE SEGMENTATION AND CLASSIFICATION".

[12] Allan Borodiny, Gareth O. Robertsz, Jeffrey S. Rosenthalx and Panayiotis Tsaparas, "Link Analysis Ranking Algorithms, Theory, and Experiments", A preliminary version of this paper has appeared in the Proceedings of the 10th International World Wide Web Conference.

[13] Hema Dubey and Prof. B. N. Roy, "An Improved Page Rank Algorithm based on Optimized Normalization Technique", (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 2 (5) , 2011, 2183-2188.

[14] Ranveer Singh and Dilip Kumar Sharma, "Enhanced-RATIORANK: Enhancing Impact of Inlinks and Outlinks", Proceedings of 2013 IEEE Conference on Information and Communication Technologies (ICT 2013).

[15] Dilip Kumar Sharma and A. K. Sharma, "A Comparative Analysis of Web Page Ranking Algorithms", (IJCSE) International Journal on Computer Science and Engineering Vol. 02, No. 08, 2010, 2670-2676.

[16] S.Ganesh, M.Jayaraj, V.Kalyan and G.Aghila, "Ontology-based Web Crawler", Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'04).

[17] Rekha Jain and Dr. G. N. Purohit, "Page Ranking Algorithms for Web Mining", International Journal of Computer Applications (0975 – 8887).

[18] Trystan Upstill, Nick Craswell and David Hawking, “Predicting Fame and Fortune: PageRank or Indegree?”, Proceedings of the 8th Australasian Document Computing Symposium.

[19] Rebecca S. Wills, “Google’s PageRank: The Math Behind the Search Engine”, Department of Mathematics North Carolina State University Raleigh, NC 27695.

[20] Ian Horrocks, “Ontologies and the Semantic Web”.