A Modified Algorithm to Handle Dangling Pages using Hypothetical Node in PageRank Computation

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

Master of Engineering
In
Software Engineering

Submitted By
Shipra Srivastava
(801031027)

Under the supervision of

Mrs. Karamjit Cheema
Lecturer
CSED

Dr. Rinkle Rani
Assistant Professor
CSED

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR UNIVERSITY
PATIALA – 147004

June 2012

Certificate

I hereby certify that the work which is being presented in the thesis entitled, "*A Modified Algorithm to Handle Dangling page using Hypothetical node in PageRank Computation*", in partial fulfillment of the requirements for the award of degree of Master of Engineering in *Software Engineering* 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 *Dr. Rinkle Rani* and *Mrs. Karamjit Cheema* 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.

Shipra Srivastava

(Shipra Srivastava)

801031027

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

Rinkle 07/06/12

(Dr. Rinkle Rani)
Assistant Professor,
CSED.

Kcheema 7/6/12

(Mrs. Karamjit Cheema)
Lecturer,
CSED.

Countersigned by

(Dr. Maninder Singh)
Head,
Computer Science and Engineering Department,
Thapar University,
Patiala.

(Dr. S. K. Mohapatra)
Dean (Academic Affairs),
Thapar University,
Patiala.

i

Acknowledgement

I would like to express my deepest appreciation to Dr. Rinkle Rani and Mrs. Karamjit Cheema, my mentor and thesis supervisor for their constant support and motivation. They had been instrumental in guiding me throughout the thesis with their valuable insights, constructive criticisms and interminable encouragement.

I am also thankful to Dr. Maninder Singh, Head, Computer Science and Engineering Department and Mr. Karun Verma, P.G. Coordinator for their constant support and encouragement.

I would like to thank all the faculty members and staff of the department who were always there at the need of the hour and provided with all the help and facilities, which I required, for the completion of this work.

I express my thanks to my family for their support and affection and for believing in me always. I also want to thank my colleagues, who have given me moral support and their relentless advice throughout the completion of this work.

Shipra Srivastava.

Shipra Srivastava

(801031027)

Abstract

The information on the web is intensifying day by day due to which there is a bulky amount of information available on the web. With this large amount of information the process of searching has become a complex task. To overcome this problem and to make the information retrieval simple there is a necessity of some efficient search engine. With the knowledge of structure of the web Google PageRank algorithm is developed. A PageRank algorithm places the most important pages at high ranks depending upon the number of inlinks linking to that page. On the web there are web pages which do not have any outlink, these pages are called dangling pages. These dangling pages are removed from computation in PageRank algorithm which is not justifiable. These pages produce the philosophical issues and computational issue. Some approaches were developed to handle these dangling nodes but none of them solved all the issues related to dangling pages. One approach to handle dangling node is developed with the help of hypothetical node. This approach solves philosophical issue related to dangling web pages, but addition of this hypothetical node to web graph increases the computational problem because the number of iteration is increased greatly.

In this thesis a modified algorithm to handle the dangling node with the concept of hypothetical node without increasing computational problem is proposed. From the modified algorithm the dangling pages are not excluded from the computation. The modified algorithm is converged in less number of iteration than existing algorithm. The quality of modified algorithm is proved by running any web graph on both the existing and modified algorithm and then the number of iteration that each algorithm take to converge the algorithm is compared.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

With the rapid development of Information Technology, information in the internet has become the main source of knowledge. With this growth rate of World Wide Web, searching in a collection of hypertext documents has received great research interest. Due to this large amount of information on the web and its growth rate locating information in such a vast data source receives a big challenge. These challenges resulted in the development of one of the most popular applications of the Internet, Search Engines. However due to the changeability and multiplicity of the information on Internet it become complex to look for required information. Presently Information retrieval on the web is much more complex than traditional Information Retrieval (IR) systems. Classical IR techniques are used by the search engine for matching the user queries to the web documents. The base of these techniques is relationship between the terms which is used in user query and the web documents. To improve the quality of the search results, search engines use many superior techniques which are generally measured by the level of user satisfaction and the relevance of the returned results to the given query. One of the most important technique is to exploit the connectivity information **(**i.e. Link Structure) hidden in the web documents to improve the search results.

Search tool used on the web must be able to discriminate the high-quality pages on the web from low-quality pages. To distinguish the high quality pages from low quality pages, variety of search engines has been developed in the field of information retrieval. For searching the information from the web people use a search engine to begin their Web activity. In this case user submits a query in the form of keywords and receives a list of Web pages that may be relevant. Because of highly volatile structure of the web number of pages on the web is increasing day by day, every day lots of pages are added, deleted and modified from the web. In the present time the order of the size of web is more than a billion pages and most of these pages contain redundant information. Therefore, there must be some method to get these redundant pages on the lower quality so that user does not waste their time in reading these pages. There must be good quality search engine so that user get their result in less time without visiting lots

of pages by getting high quality pages in the top of the result. The Web on the other hand is massive, less coherent, changes more rapidly and spread over geographically distributed computers. This requires new techniques or extensions to the old ones to deal with the gathering of the information, to make index structures scalable and efficiently updated and to improve the discriminating ability of search engines. There are lots of search engine used in the world like Google, yahoo and Ask etc. These search engines provide the user relevant pages regarding their search in the result.

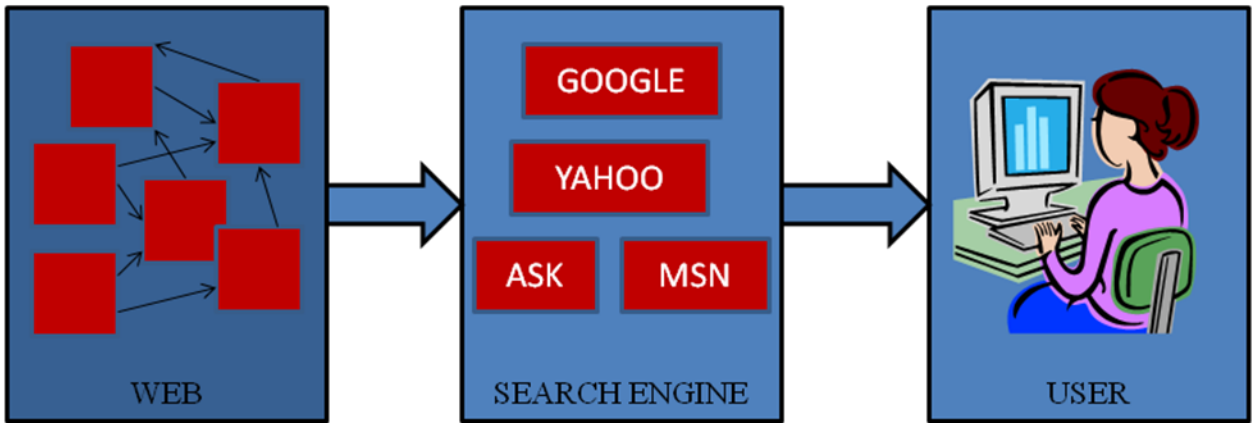


Figure 1 : Searching the Web through Search Engine

1.1 Web Mining

With the increasing growth of the amount of data on the web it has become gradually more difficult for users to search and use information and for content providers to classify and catalog documents present on the web. To overcome these problems researchers are working on automated methods to work with web documents so that they can be more easily browsed, organized and cataloged with minimal human intervention. Web mining process is used to find out relevant and hidden information from this large data set [47] [34]. According to Oren Etzioni "*Web mining is the use of data mining techniques to automatically discover and extract information from World Wide Web documents and service*" [15]. The data on the World Wide Web (WWW) continues to grow in both size and complexity which makes the application of web mining necessary to obtain accurate and relevant information from the web in less time and less complexity. The characteristic feature of web mining is the use of data mining techniques to

elaborate on web content (text, images and records etc), web structure (hyperlinks, tags etc) and web usage (http logs, app server logs, etc) of web resources. Web Mining is an important process of transformation of data from human understandable content to machine understandable semantics. It discovers the intrinsic relationships among Web data then expressed in the forms of textual, linkage or usage information via analyzing the features of the Web and web based data using data mining techniques.

1.1.1 Taxonomy of Web Mining

Kosala and Blockeel classified web mining into three parts according to the kinds of data to be mined [25]:

1) Web Content Mining

2) Web Structure Mining

3) Web Usage Mining

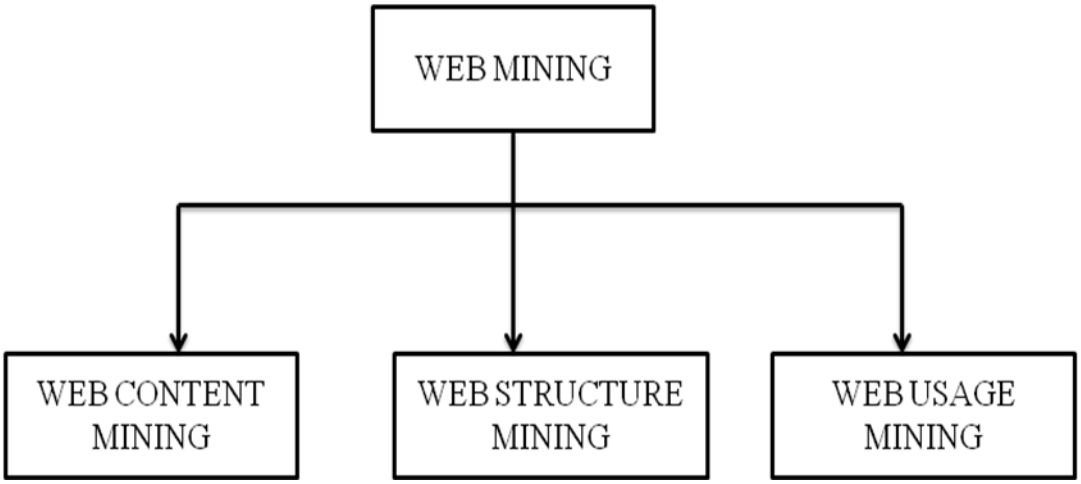


Figure 2: Taxonomy of Web Mining

1. Web Content Mining: The process of extracting useful information from the contents of web documents is known as web content mining. Margaret H. Dunham [13] stated web content mining can be thought of the extending the work performed by basic search engines. Collection of facts by which a Web page is designed to convey to the users represents the content data of

the web. This content data consist of text, images, audio, video or structured records such as lists and tables. It is a form of text mining that takes advantage of the semi-structured nature of the web page text caused by HTML tags or XML markup [22]. Text mining and its application to web content has been the most widely researched area. Some of data on the web are semi structured such as HTML documents and some are more structured data like data in the tables or database generated HTML pages, but most of the data on the web is still unstructured text data and these unstructured data is the main challenge for web content mining.

2. Web Structure Mining: The structure of the web is based on a graph with about million of web pages and billion of hyperlinks. Web structure means connected network between web pages. The web consists not only of pages but also of hyperlinks pointing from one page to another. The web graph consists web pages as nodes and hyperlinks between these web pages are represented by the edge of the graph. Web structure mining is the process of discovering structure information from the Web [35]. Based on the topology of the hyperlinks, web structure mining will categorize the web pages and generate the information such as the similarity and relationship between different web pages. Web structure mining can be further divided into two kinds based on the kind of structural data used.

(i) **Hyperlinks**: For connecting a web page either within the same web page or to a different web page a structural unit is used which is called Hyperlink. Hyperlink is also classified into two parts [40] :

(a) The hyperlink that connects to a different part of the same page is called an Intra-Document Hyperlink.

(b) The hyperlink that connects two different pages is called an Inter-Document Hyperlink .

(ii) Document Structure: The web contents within a web page can also be organized in a tree-structured format based on the various HTML and XML tags within the page [40]. Mining efforts here have focused on automatically extracting Document Object Model (DOM) structures out of documents.

On the web when an author of any web page creates a hyperlink pointing to another web page then this can be considered as that author's support of the other page. When different author give

their support to a given page on the web then it indicate the importance of the page and this lead to the discovery of high quality web pages. For this process two main algorithms which used are:

(i) HITS (Hyperlink-Induced Topic Search) Algorithm

(ii) Google PageRank Algorithm

(i) HITS Algorithm – For rating the web pages on the web Hyperlink-Induced Topic Search (HITS) which is link analysis algorithm is used. The concept of hub and authority is introduced in HITS algorithm, where hubs is pages that refer to many other pages and authorities is pages that are referred by many other pages. The idea behind Hubs and Authorities originated from the creation of web pages. During the design of web pages certain web pages were stated as hubs. These pages did not carry any authoritative information but were used to serve as large directories of information that led users directly to other authoritative pages. In other words a good hub represented a page that pointed to many other pages and a good authority represented a page that was linked by many different hubs [36]. Hubs and Authorities can be viewed as 'fans' and 'centers' in a bipartite core of a Web graph [45]. The nodes on the left represent the hubs and the nodes on the right represent the authorities as shown in the figure 3.

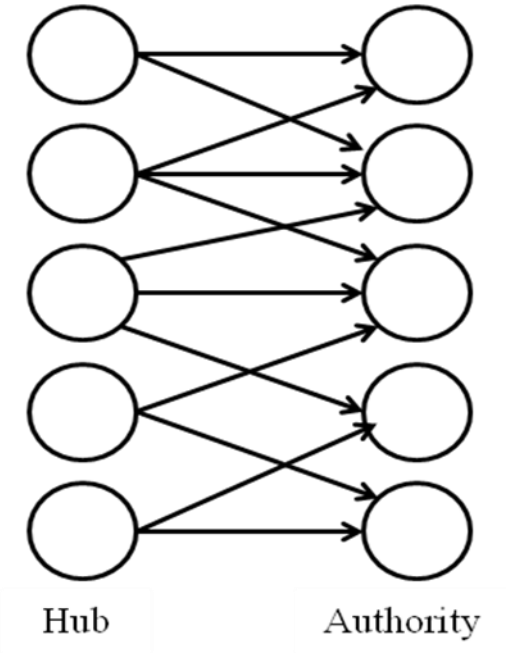


Figure 3: Bipartite Core

(ii) Google PageRank Algorithm – For measuring the importance of pages on the web PageRank algorithm is used by the Google. PageRank algorithm can be defined as a link based probabilistic algorithm. This algorithm assigns a numerical value to each page in the World Wide Web so relevant importance of each page rank with other pages on the web can be measured when searching is performed. PageRank algorithm is developed by Google founders Sergey Brin and Larry Page [9] and they use this PageRank algorithm use as a foundation for their search engine. After applying the algorithm Pages that are deemed more important will move up in the result of the pages of a user's search when a user enters their query in the search box. PageRank algorithm finds the pages on the web that matches the user's query and lists those pages in the order of their PageRank. When page rank of any page is calculated then it take into account page rank of each page that point to that page and also the outdegree of each page point to it.

3. **Web Usage Mining:** Web usage mining is the process to discover interesting usage patterns from web data in order to understand and better serve the needs of web-based applications. The identity or origin of web users along with their browsing behavior at a web site is captured by Usage data. Every user leaves a path by their sequence of the web pages that they have accessed when they visit any web site. These records are often collected by a logging algorithm on the web server and from this information web usage mining tries to discover the useful information such as navigation patterns [44]. Web usage mining focuses on the techniques that could predict user behavior while the user interacts with the web. Most existing web analysis tools [49] provide mechanisms for reporting user activity in the servers and various forms of data filtering. Web usage mining can be combined with the other techniques in order to detect frequently used paths. Web usage mining itself can be classified further depending on the kind of usage data considered.

(i) Web Server Data
(ii) Application Server Data
(iii) Application Level Data

1.2 Search Engine

Search engines are program that search for specified keywords given by user from the documents and returns a list of the documents where the keywords were found [42]. Normally it is used to refer to large web-based search engines that search through billions of pages on the internet like Google, yahoo etc. An Internet based tool that searches an index of documents for a particular term, phrase or text specified by the user. Many of the search engines use well-known information retrieval (IR) algorithms and techniques. Initially IR algorithms were developed for relatively small and coherent collections of news- paper articles or book catalogs in a (physical) library but now for searching in this geographically distributed computers there is need of new techniques or extensions to the old ones to deal with the gathering of the information, to make index structures scalable and efficiently updated and to improve the discriminating ability of search engines. In the search engine

(i) Crawler: Retrieves the web pages and web contents.
(ii) Indexer: Stores and indexes information on the retrieved pages.
(iii) Ranker: Measures the importance of Web Pages.
(iv) Retrieval Engine: Performs lookups on index tables against query.

Search engine is different from web directories which are maintained only by human editors. By running an algorithm on a web crawler search engines maintain real time information.

1.2.1 Architecture of Web Search Engine

The architecture of a web search engine contains a front-end process and a back-end process. In the front-end process the user first enters the search words into the search engine interface. Search engine interface is usually a web page on the internet with an input box. In this input box user enter their query or keyword. The application then parses the search request through the query parser into a form that a search engine can understand and after that search engine executes the search operation on the index files. The result returned by search engine than ranked. After ranking the search engine interface returns the search results to the user. Because of the ranking the user get high quality pages at the top in the result list. In the back-end process,

a spider or robot fetches the Web pages from the Internet and then the indexing subsystem parses the Web pages and stores them into the index files [1]. Google, Excite, Lycos, AltaVista, wInfoseek, and Yahoo are example of search engines. They index million of sites on the Web so that web surfers can easily find web sites with the desired information. Search engines can locate relevant web sites when users perform search by creating indexes or large databases of web sites (based on titles, keywords and the text in the pages). If the Boolean operators are used for searching then more relevant sites is easily listed in the result. The final architecture of web search engine is shown in Figure 4.

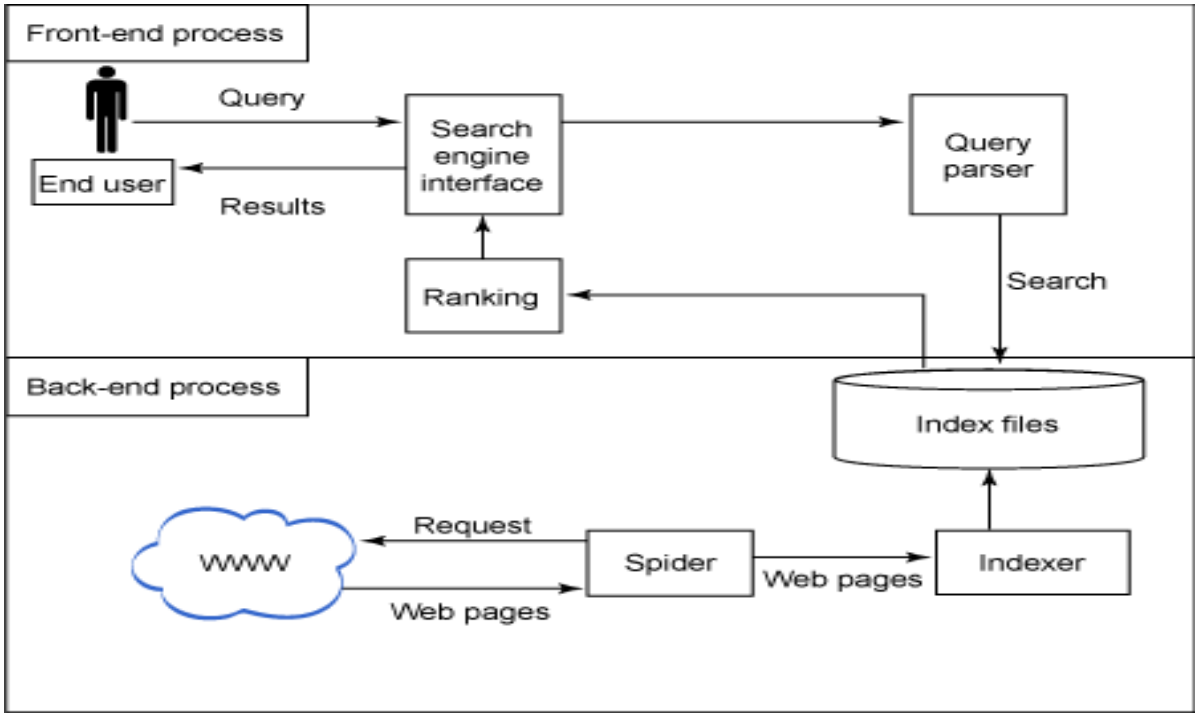


Figure 4: Architecture of Web Search Engine [15]

1.3 Need of Search Engine Ranking

An enormous amount of information is contains in the World Wide Web. Anyone can easily get information regarding their search but to locate resources that are both high in quality and relevant in this large amount of information create difficulty for users. According to Google the numbers of web pages are increasing every year. The growth in web pages from the last two year is shown in figure 5.

From the figure 5 it is clear that from the May 2010 to May 2012 the number of web pages on the web is increased from 14 billion to 56 billion. From this large container of web pages users want the relevant result or high quality pages relevant to search, so ranking is very important for search engine. It is the search engines ranking that finally bring your website to the notice of the prospective customers. When a topic is typed for search instantly the search engine will sort through the millions of pages it has indexed about and the high quality pages are presented at the top in the list. The matched searches are also ranked so that the most relevant ones come first. Search engines essentially act as filters for the wealth of information available on the Internet. Ranking allow users to quickly and easily find information that is of genuine interest or value to them without the need to wade through numerous irrelevant web pages. There is a lot of filtering to do in 2004 the number of pages in Google's index exceeded the number of people of the planet, reaching the staggering figure of over 8 billion [21]. With that much information the Internet would be essentially unusable without the Search Engines.

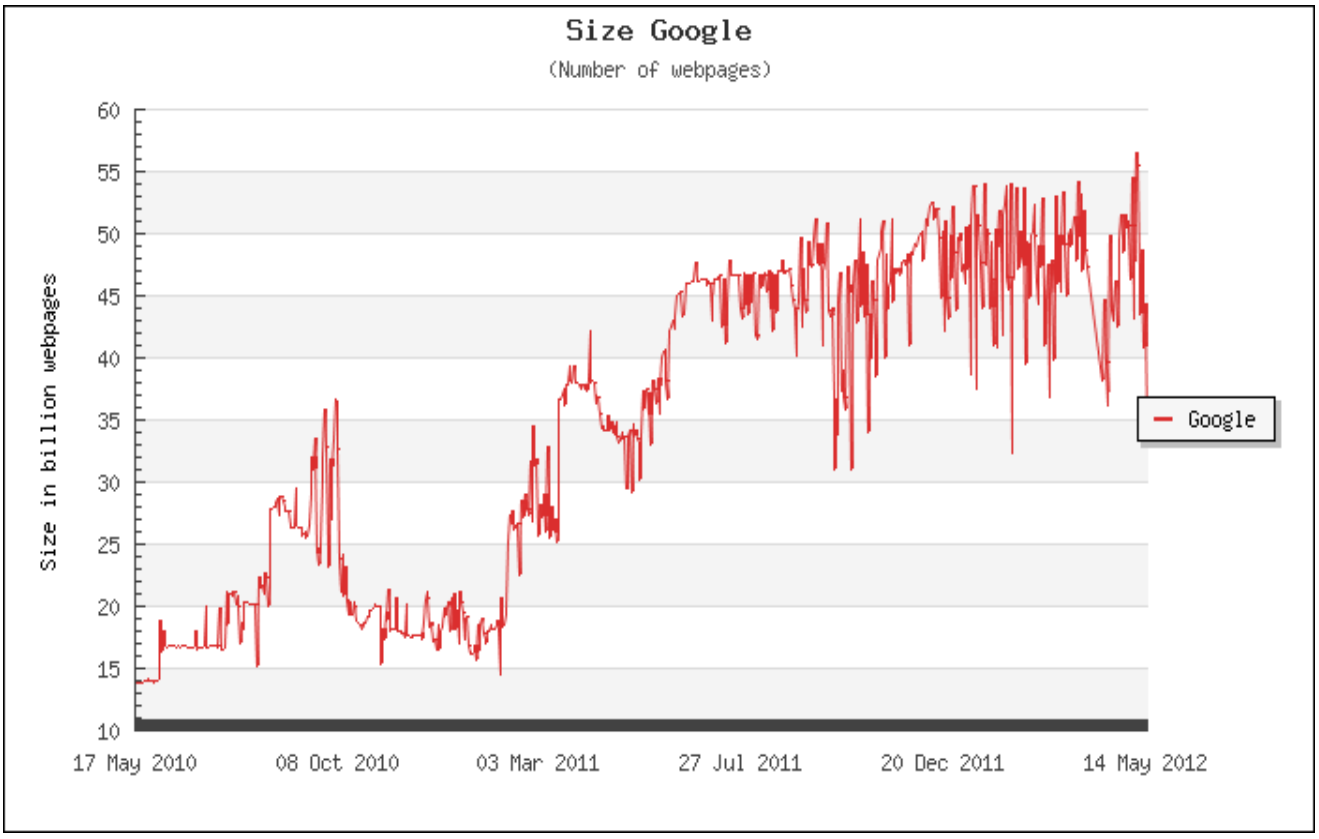


Figure 5: Growth Rate of Web Pages on the Google in Last Two Years [46]

The goal of the Search engines ranking is to provide users with search results that lead to relevant information on high quality websites. Complex algorithms are used to assess websites and web pages by Search engines and assign them a ranking for relevant search phrases. These algorithms are frequently updated. Google looks at over 200 different metrics when assessing websites, including copy, in-bound links, and website usability and information architecture [21].

1.4 Google Search Engine

Google is one of the best and most popular website in the world. It is an undeniable giant search engine on the Internet world since its launching in the year of 1998. One of the reasons why Google is such an effective search engine is it's PageRank algorithm developed by Google's founders Larry Page and Sergey Brin [9]. Google use PageRank algorithm as a basis for their search. Page rank is determined entirely by the link structure of the World Wide Web. According to global market share statistics, Google is far ahead than any other search engine in terms of usage and content searching including all the countries in the Internet world. Apart from Yahoo no other search engine is near to the mark of 100M unique visitors' per month. Google combines Image Search Engine and Video Search Engine capabilities in its web content searching criteria [39]. Google remains the top and best search engine in the world till this date only because of quality search. Google does maintain its Alexa Rank as No.1 today with the title of "Best Search Engine of the Year-2012" [39]. Due to PageRank algorithm Google's rise to success was in large part. PageRank algorithm assigns rank to web pages so that high quality page receive high page rank and they are listed at the top when searching is performed by the user. User gets relevant information regarding their search without visiting lots of website in less time. When Google was a Stanford research project it was nicknamed BackRub because the technology checks back links to determine site's importance. The PageRank algorithm analyzes human generated links assuming that web pages linked from many important pages are themselves likely to be important. The algorithm computes a recursive score for pages based on the weighted sum of the page ranks of the pages linking to them. PageRank is correlates well with human concepts of importance. Ranking places very important role for any search engine and it is also useful for the user to get high quality pages. There are many other secret criteria added for determining the ranking of pages on result lists in addition to PageRank. Reported to be over 250

different indicators, the specifics of which are kept secret to keep spammers at bay and help Google maintain an edge over its competitors globally [3].

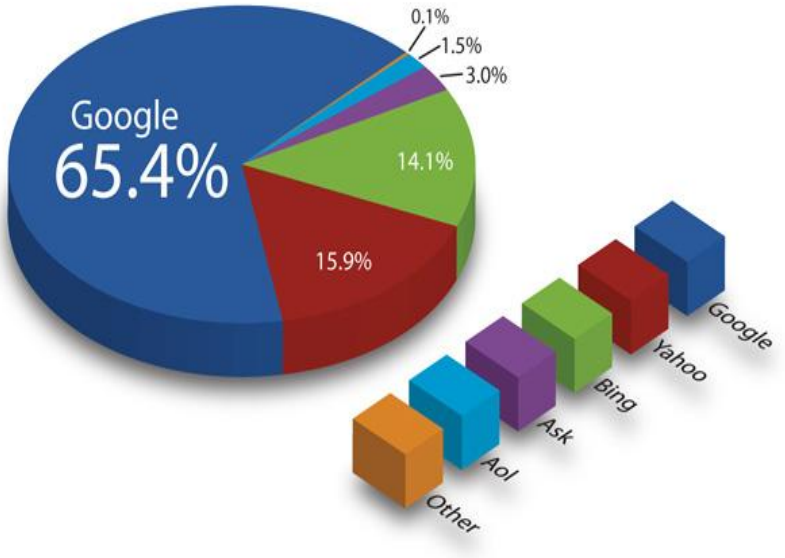


Figure 6: Search Engine Visitor Data in 2011 [3]

1.5 The Hyperlink Structure of the Web

The hyperlink structure of the web is represented by the web graph with the web pages are presented as node and links between these pages are presented as edges. The World Wide Web contains billion pages and these pages are represented in such a nodes and edges form. It is called a Hyperlink graph.

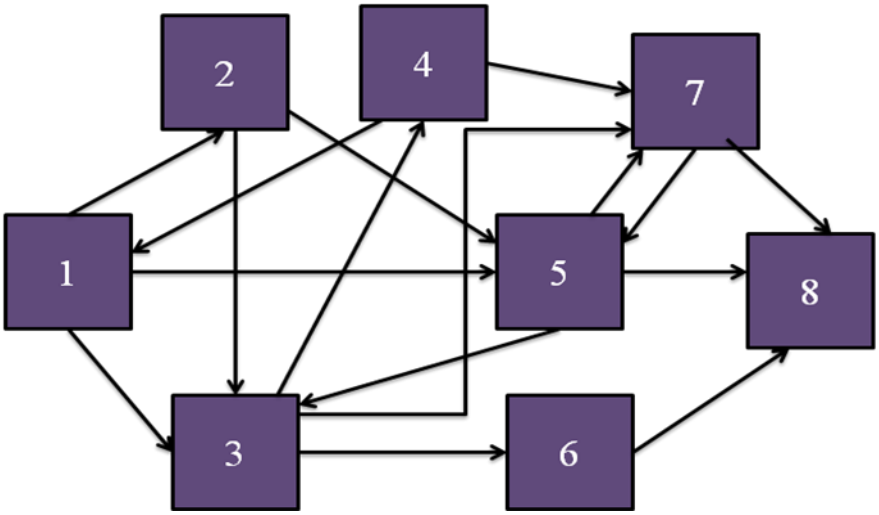


Figure 7: Hyperlink Structure of the Web

These hyperlink graphs contain great source of latent information. In most of the links based analysis algorithms this hyperlink structure in the form of graph is used as a basis for implementing the algorithm. All links in the web graph have some weight but it is not compulsory that all the weights are same.

In the figure there are 8 nodes which represent the web pages and links between these pages shows the hyperlinks between these pages. For user who wants to retrieve efficient information from the web use this hyperlink graph for retrieving the information. Successful search engines like Google depend heavily on link analysis and it also uses other information such as text, anchor text, etc. along with a link based ranking scheme.

In this thesis in chapter 2 which is Literature review the PageRank algorithm and other Link based algorithm are discussed. The concept of dangling pages are also discussed in this chapter 2.Chapter 3 contain the problem statement and chapter 4 contain the proposed modified algorithm. Chapter 5 presents the experiment results to show that proposed algorithm can solve the problem defined in chapter 3.

Chapter 2

Literature survey

2.1 The History of PageRank

PageRank algorithm is developed by Google founders Surgey Brin and Larry Page [9] at Stanford University in 1998 and they use PageRank algorithm as a foundation for their search engine. Initially the project began as a Web project named BackRub and utilized the link structure of the Web. In 1995 Larry Page visited the computer science department of Stanford University Larry Page and there he met Sergey Brin. Larry Page began working on a Web project initially called BackRub which exploited the link structure of the Web. When Sergey Brin found Page's work on BackRub interesting then they started working together on a project that permanently change web search. After some time Brin and Page realized that they were creating a search engine that adapted to the ever increasing size of the Web so they replaced the name BackRub with Google. Soon they began realizing that they were actually making the backbone of a strong search engine. After developing their PageRank algorithm both of them tried to convince other companies which use existing search engine about the effectiveness of their algorithm [41]. But the other companies were not ready to use their algorithm so Larry and Surgey started a company named Google Inc. in September 1998 which used the PageRank algorithm as the basis for a search engine.

2.2 Overview of PageRank Algorithm

PageRank algorithm is Google's method of measuring a page's importance. PageRank algorithm is link based probabilistic algorithm. It assigns a numerical value to each page in the World Wide Web so that it measures the relevant importance of each page rank with other pages on the web. Pages that are deemed more important will move up in the result of the pages of a user's search when a user enters their query in the search box. PageRank algorithm finds the pages on the web that matches the user's query and lists those pages in the order of their page rank. When the page rank of any page is calculated the number of back links to that page and also the number of out-links of each page that point to it are taken into account. The core of the PageRank algorithm

involves computing the PageRank vector, which is the stationary distribution of the so-called Google matrix and a measurement of the importance of the web pages [20].The dimension of the Google matrices exceeds 11.5 billion so only a small set of algorithms for computing its stationary distribution can be applied [21].

The Ranking algorithm is used to distinguish high quality pages from low quality pages. When user performs the searching then the search tool generally views only the top few pages returned by the search engine. Ideally, a search engine should be ranked by query relevance so that high quality and good quality page must be on the top of search. Links provide more complete and concise information about documents than text contained in the actual document when Brin and Page stress that anchor text should be associated both with the originating page and the linked page. This method also allows Google to perform searches on documents that cannot be indexed by a text-based search engine, like images, programs, and databases [8].

For an information retrieval system determining a page's relevance to query terms is a complex problem but the inherent hyperlink structure of the web may be used to generate an approximation. The idea of using the hyperlink structure of the web idea is implemented in the link based algorithms. The link based algorithms depend on the relationship among objects in a set and not on the content of the elements within the set. Probabilistic algorithms are algorithms which return the most probable value of the result for the same set of inputs. PageRank algorithm has graph theory as its backend for theoretical formulation and matrix theory as its front end for its implementation. **Google PageRank** is probably one of the most important algorithms ever developed for the web. PageRank algorithm is used by Google to determine best search results and to keep the search clean and efficient. The sorting of the search results is done by PageRank algorithm. Because PageRank algorithm adopts citation analysis theory and sorts the search results by setting the importance of pages. It defines two standards to weigh PR (Page Rank) value of the pages. The higher of the PR value, the more forward of the sorting result: (1) If more number of the hyperlink points to a page, more is importance of that page and higher is the PR value (2) If there is an important hyperlink of the page pointing to another page, the another page is also important and PR value is high. From the figure 8 it can be seen that page 1 has a

hyperlink pointing to page 4 and page 2 and page 4 has also a link from the page 3 and 5. The algorithm considers that page 1 vote page 2 and page 4 to show that page 2 and 4 are valued link. Page 3 and page 5 also vote's page 4. It means Page 4 is an important page. Depending on the number of hyperlinks and the importance of the source page that points to page 4 the importance of page 4 is judged and PR value is assigned to page 4. If the PR value of page 1, page 3 and page 5 is high, the page 4 will share some PR value from page 1, 3 and 5.

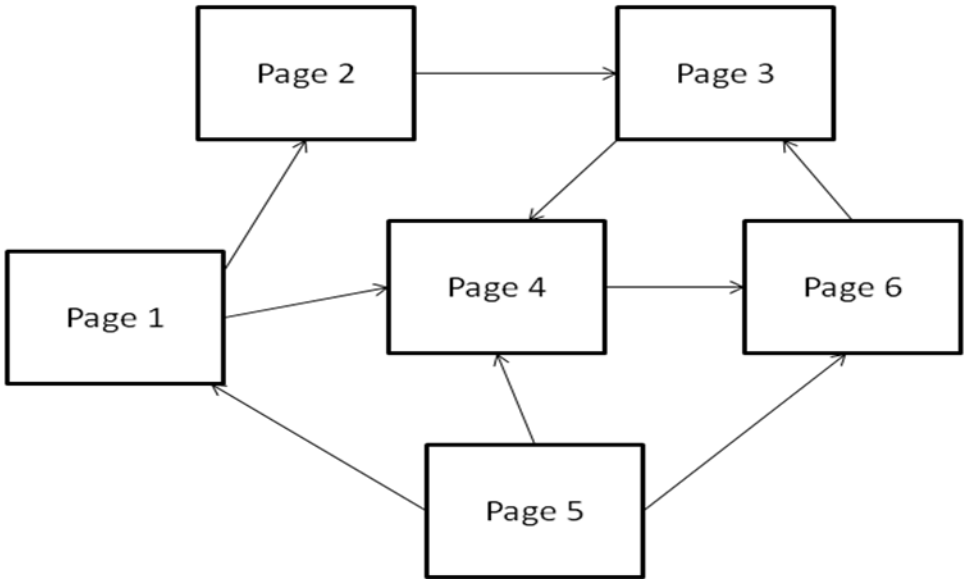


Figure 8: A Directed Web Graph with 6 Web Pages

2.2.1 Simplified Version of PageRank

Let u, v be web pages and Bu be the set of pages that point to page u. Further let Nv be the number of links from v. Let c < 1 be a factor for normalization [22]. A simple ranking R is defined, which is a simplified version of PageRank:

$$R(u) = c \sum_{V \in Bu} \frac{R(v)}{Nv} \quad (1)$$

The rank of a page is divided among its forward links evenly to contribute to the ranks of the pages they point to. The equation is recursive. But some problem is associated with this simplified PageRank version .The problem arises when two Web pages point to each other but

not to any other page while some other web page points to one of them.In this case a loop will be generated during the iteration. This loop will accumulate the rank but will never distribute any ranks. These traps formed by loops in a graph without out edges are called rank sinks.

2.2.2 Random Surfer Model

It is a hypothetical user who chooses a webpage at random from the available set of web pages and then continue to select a random link on that webpage and surfs to that webpage. This process is repeated indefinitely. The choice of which webpage to visit next does not depend on the previous webpage [32]. While surfing the Web, user move from one page to other page by randomly choosing an outgoing link from one page. But sometimes this process lead to dead end, meaning, to a page that has no outgoing links or cycles around cliques of interconnected pages. So certain fraction of the time is wasted in simply choose a random page from the Web .This theoretical random walk is known as a Markov chain or Markov process [38]. The limiting probability that an infinitely dedicated random surfer visits any particular page is its PageRank. A page has high rank if other pages that points to this page have high page rank. A model of a web surfer has to be created to avoid rank sinks. This surfer keeps clicking on hyperlinks at random. The task is modeling the behavior of the surfer who periodically jumps to a random page. Therefore let E (u) be a vector over the web pages that correspond to a source of ranks. The random surfer chooses a page based on a distribution in E. Now PageRank can be defined as an assignment R´ to a Web page which satisfies the following formula such that c is maximized and the L1 norm of R´ = 1 (convergence criteria):

$$R'(u) = c \sum_{V \in Bu} \frac{R'(v)}{Nv} + c\,E(u) \qquad (2)$$

Random surfer will encounter two problems while doing the random surfing. First, random surfer will find out it keep revisits the same cycle of node and get stuck in a loop. In the figure 8 random surfers will revisit the same cycle formed by node 1, 2 and 3 never get a chance to visit node 4. The second problem that a random surfer will encounter is when arriving at a dangling node. Dangling node is a node without out links and this dangling node will get random surfer to

stuck. From figure 9 it can be seen that the random surfer will get stuck after arriving at node 4 as node 4 does not have any outlink.

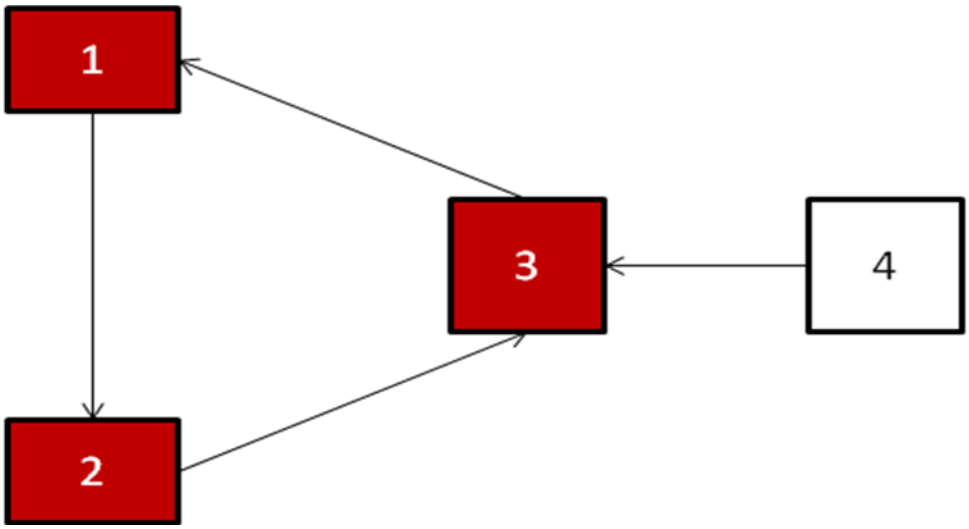


Figure 9: Simple Loop which act as Cycle

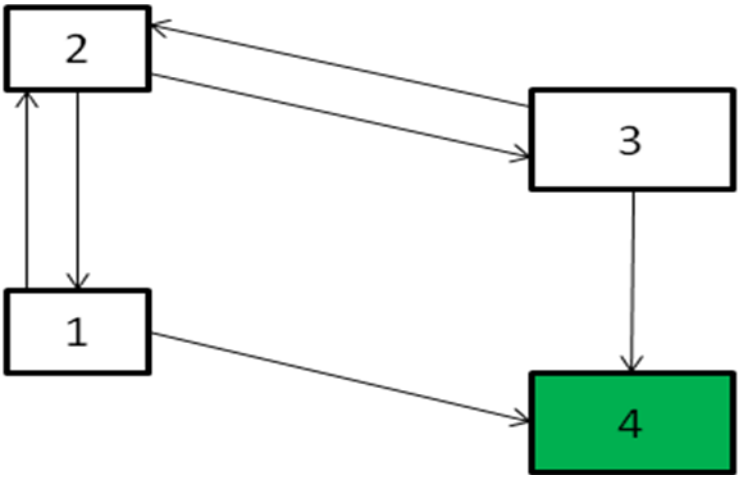


Figure 10: A Directed Web Graph with Dangling Page 4 (Green node)

2.2.3 The mathematical formula for calculating the PageRank

The PageRank formula is described by Lawrence Page and Sergey Brin [9]. The formula of PageRank is

$$PR(A) = (1-d) + d\,(PR(T1)/C(T1) + \ldots + PR(Tn)/C(Tn)) \quad (3)$$

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.

• d is a damping factor which can be set between 0 and 1 usually 0.85.

PageRank is determined for each page individually but it does not rank web sites as a whole. Further, the PageRank of any web page is recursively defined by the PageRank of those pages which link to computing page. The PageRank of pages Ti which link to page A does not influence the PageRank of page A uniformly. Within the PageRank algorithm, the PageRank of a page T is always weighted by the number of outbound links C (T) on page T. This means that more the outbound links a page T has, the less will page A benefit from a link to it on page T. The weighted PageRank of pages Ti is then added up. The outcome of this is an additional inbound link for page A which will always increase page A's PageRank. Finally the sum of the weighted PageRank of all pages Ti is multiplied with a damping factor d and the value of d is normally set to 0.85.

2.3 Damping Factor

In the random surfer model where user jumps from a given vertex to another random vertex in the web graph, then probability of this jump into the PageRank model is incorporated by the damping factor D. According to the concept of damping factor, constant d is used to solve the problem of cyclic and dangling node problem which is discussed in figure 9 and 10. The probability that will help random surfer at any step to continue their traversing still there is not any outlink from the last position to any another node not which is not in path is called Damping factor. It is also known as teleportation factor. The damping factor is subtracted from 1 and divided by number of nodes, N for normalization. Normally the value of damping factor will be set to 0.85 but different values of damping factor have been studied by various studies [2,5,50] .For example suppose d = 0.85, then 85% of the time the random surfer follows the hyperlink structure and the remaining 15% of the time the random surfer will teleport to a new page. The value of damping factor parameter is also very sensitive for the convergence of PageRank

calculation. The value of damping factor value cannot be set too high or too small because it will take longer time to reach convergence of page rank algorithm. Since there is little damping (1 - d) PageRank received from external pages will be passed around in the system. The PageRank algorithm converges earlier if the value of d is too small but the relative PageRank will be determined by PageRank received from external pages rather than the internal link structure [29]. The effect of d on expected number of power iteration is shown in Table 1 as follows:

Table 1: Effect of d on expected number of power iteration [29]

d	Number of Iterations
.5	34
.75	81
.8	104
.85	142
.9	219
.95	449
.99	2,292
.999	23,015

2.4 Convergence Criterion

The page rank value is stable when sum value of difference in PageRank value reaches zero, $\sum$ || PRi+1- PRi || == 0 where each iteration is denoted by value of i. $(i+1)^{th}$ means next iteration and i^{th} is previous iteration. It means the algorithm is converged when sum of difference in the page rank value at the $(i+1)^{th}$ iteration and at the i^{th} iteration is Zero. The social graph is full of cyclic and dangling problem discussed above so it will quite difficult to reach zero in real world implementation. For overcoming this problem the termination criteria of iteration is setup to some tolerance, t value $\sum$ || PRi+1- PRi ||< t. For Google, t =0.001 is used for calculating PageRank. For Google's ranking this degree of accuracy is apparently adequate. The experiment

is conducted by Haliwala and the result shows that the exact value of PageRank are not important as the correct rank ordering [18]. This whole experiment is performed to prove Google decision for t value setting. Rank ordering between two rankers can be compared using a measure such as Kendall's Tau [16].

2.5 Transition Probability Matrix

For calculating the PageRank, the calculation is begin by building a mathematical model of the link structure of the web. The transition probability matrix M [28] is constructed by defining the elements of a matrix M as

$$M_{ij} = \begin{cases} 1/|O_i| & \text{, if there exists a link from page i to page j} \\ 0 & \text{, otherwise} \end{cases} \quad (4)$$

Where Oi is the outlink degree of node i. For computing the page rank vector the matrix need to be stochastic because Markov chain is only defined for stochastic matrix.

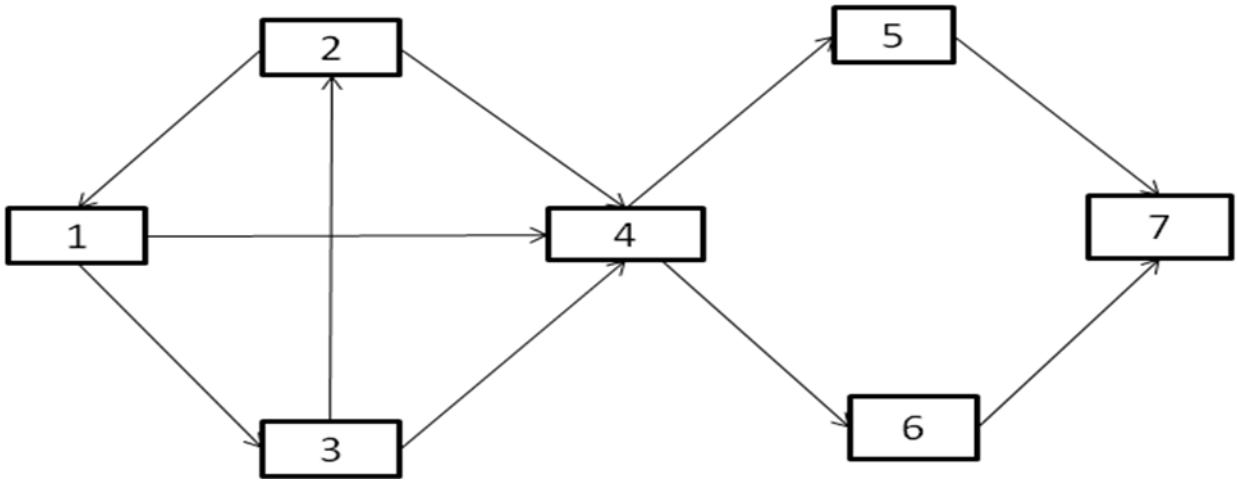
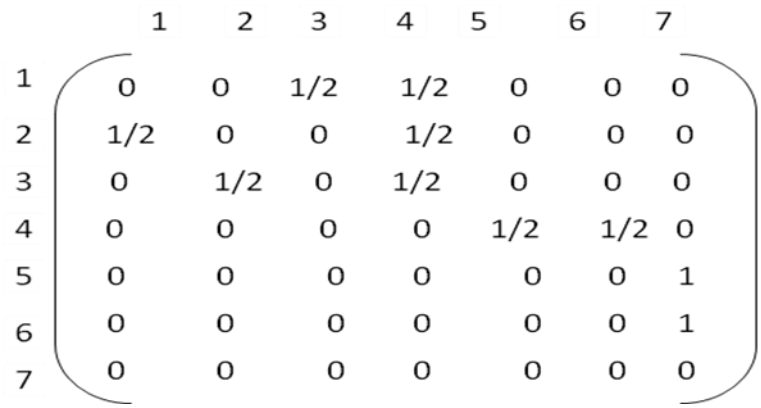


Figure 11: A directed Web Graph with 7 Web Pages



	1	2	3	4	5	6	7
1	0	0	1/2	1/2	0	0	0
2	1/2	0	0	1/2	0	0	0
3	0	1/2	0	1/2	0	0	0
4	0	0	0	0	1/2	1/2	0
5	0	0	0	0	0	0	1
6	0	0	0	0	0	0	1
7	0	0	0	0	0	0	0

Figure 12: Transition Probability Matrix of Web Graph having Seven Node

2.6 PageRank Algorithm

The original Page rank algorithm calculates the Page rank of each node present in the web graph. The algorithm is based on the mathematical formula described in equation 3. The PageRank algorithm is described by Larry Page and Surgey Brin in [9]. The implementation of this algorithm is described in [12].The complexity of that implementation is O (k ×n× m) Where k =number of iterations that algorithm take to converge, n = $|V(G)|$ or total number of vertexes, m = $|E(G)|$ or total number of edges. One way to improve this would be to cycle through all edges instead, remembering the added PageRank for each vertex and the number of outgoing links of each vertex**.** When performing this method the complexity is reduced and the complexity is O(k× m + n), where k =number of iterations, n = $|V(G)|$ and m = $|E(G)|$ which is a lot better, since the transition matrix of the web's underlining link-graph is very sparse (each page has very little links to other pages)[12].

After applying the original page rank algorithm on the web graph shown in figure 8, the page rank value of each node is shown in figure 13.The algorithm takes 136 iteartion to converge the algorithm.

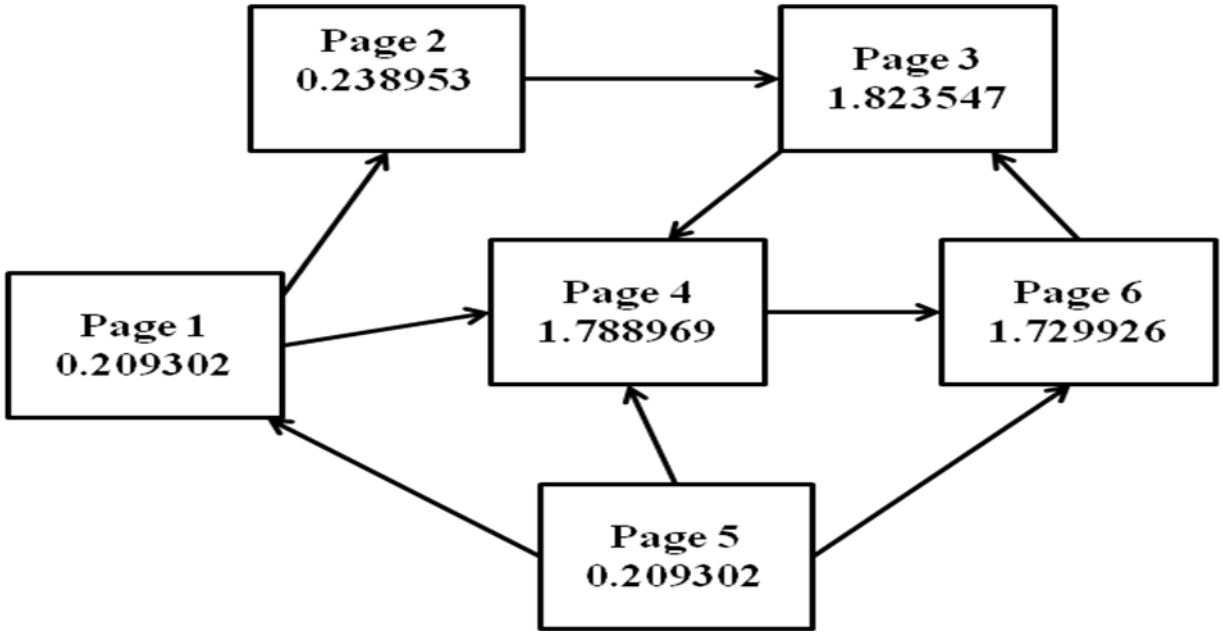


Figure 13: Web Graph along with the Page Rank value of each Node

1:0.1499999765 0.1499999765 0.1499999765 0.1499999765 0.1499999765 0.1499999765

2:0.1924999702 0.2137499681 0.4049999431 0.3412499514 0.1924999702 0.3199999542

3:0.2045416357 0.2318124664 0.6036874227 0.5760624276 0.2045416357 0.4946041028

.....

.....

70:0.209302295 0.2389534543 1.8235241702 1.7889459338 0.209302295 1.7299030748

71:0.209302295 0.2389534543 1.8235280728 1.7889490421 0.209302295 1.7299063811

.....

.....

132:0.209302295 0.2389534543 1.8235473861 1.7889687753 0.209302295 1.7299257965

133:0.209302295 0.2389534543 1.8235473861 1.788968776 0.209302295 1.7299257965

134:0.209302295 0.2389534543 1.8235473861 1.788968776 0.209302295 1.7299257972

135:0.209302295 0.2389534543 1.8235473868 1.788968776 0.209302295 1.7299257972

136: 0.209302295 0.2389534543 1.8235473868 1.788968776 0.209302295 1.7299257972

Figure 14: Result of Applying the PageRank Algorithm on Web Graph shown in figure 8

From the figure 14 it can be seen that the value of all nodes at the iteration 134 and 135 is same expect the node 3 so the algorithm is not converged but sum of difference between the values at 135^{th} and 136^{th} iteration is 0, so the algorithm is converged here.

2.7 Other Algorithms to Rank Web Pages

A number of techniques which is based on the structure of hyperlinks have been developed to rank the web page. The PageRank algorithm is used for Ranking and the two other algorithms which is used is HITS algorithm, proposed by Kleinberg [24] and the SALSA algorithm.

2.7.1 The INDEGREE Algorithm

The INDEGREE algorithm is predecessor of all link analysis algorithms. It is a simple heuristic to rank the web pages present according to their popularity (often also referred to as visibility) [37]. The popularity of a page is measured by the number of inlinks to that page. Because this algorithm ranks the pages according to their in-degree in the web graph so it is referred as the INDEGREE algorithm. For every node i,

$$A(i) = \frac{B(i)}{E}$$

Where B(i) determines the number of backward links of page and E represent the total number of edges in hyperlink graph. In the early days of web search this simple heuristic INDEGREE algorithm was applied by several search engines. According to Kleinberg [24], the INDEGREE algorithm is not convenient enough to capture the authoritativeness of a node, even when restricted to a query dependent subset of the Web. The concept that INDEGREE algorithm uses is that a good authority is a page that is pointed by many nodes in the graph. Brin and Page [9] extended the concept used in INDEGREE algorithm by observing that not all links carry the same weight. When any pages have a link from high quality pages then the page should confer more authority. So it is not only important that how many pages point to a page but also the quality of these inlinks pages are also important. So after the INDEGREE algorithm the

PageRank algorithm was proposed in which we consider the importance of web pages that link to other page when ranking the web pages present on the web.

2.7.2 Hyperlink-Induced Topic Search (HITS) Algorithm

HITS algorithm also known as hubs and authorities is a link analysis algorithm. Kleinberg [24] proposed a more refined notion for calculating the importance of web pages which was independent of Brin and Page. According to Kleinberg it was not necessary that good authorities always point to other good authorities. Instead, there were special nodes that act as hubs that contain collections of links to good authorities. It was a precursor to PageRank. Kleinberg proposed a two-level weight propagation scheme where endorsement conferred on authorities through hubs, rather than directly between authorities. Every page can be thought of as having two identities

- Hub - Captures the quality of the page as a pointer to useful resources.
- Authority - Identity captures the quality of the page as a resource itself [7].

If two copies of each pages are made, then graph G can be visualized as a bipartite graph, where hubs point to authorities. There is a mutual reinforcing relationship between the two. A good hub is a page that points to good authorities, while a good authority is a page pointed to by good hubs.

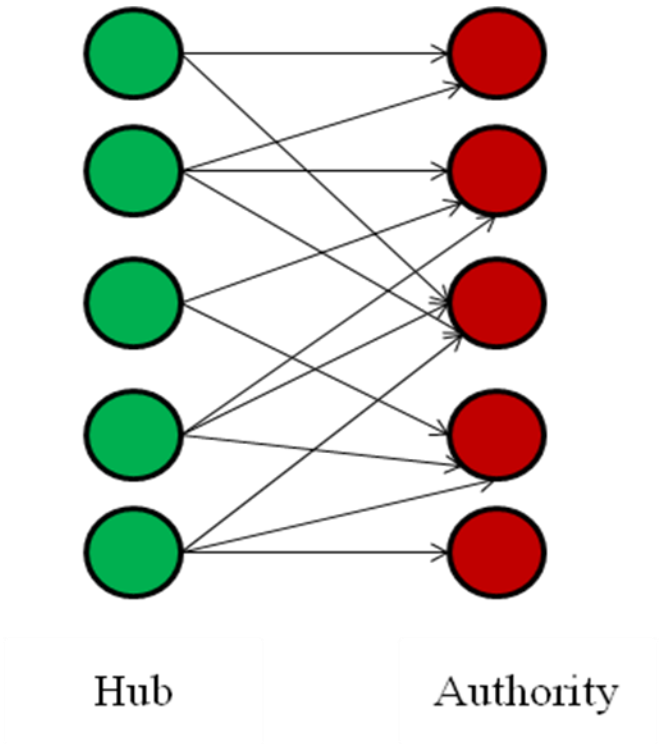


Figure 15: Hub and Authority Links

The premise of the algorithm is that a web page serves two purposes: to provide information on a topic, and to provide links to other pages giving information on a topic. This gives rise to two ways of categorizing a web page. First a web page is considered to be an authority on a subject if it provides good information about the subject. Second, the web page is considered to be a hub if it provides links to good authorities on the subject. The HITS algorithm is an iterative algorithm developed to quantify each page's value as an authority and as a hub. In other words a good hub represents a page that points to many other pages and a good authority represents a page that is linked by many different hubs. HITS assigns importance scores to hubs and authorities. The idea behind Hubs and Authorities stemmed from a particular insight in the creation of web pages when the Internet was originally forming that is certain web pages known as hubs served as large directories that were not actually authoritative in the information that it held but were used as compilations of a broad catalog of information that led users directly to other authoritative pages [19].

The HITS algorithm is not applied to the graph representing the whole web but rather to a sub-graph, typically of 1000-5000 nodes, derived from traditional text matching of the query terms in

the search topic.

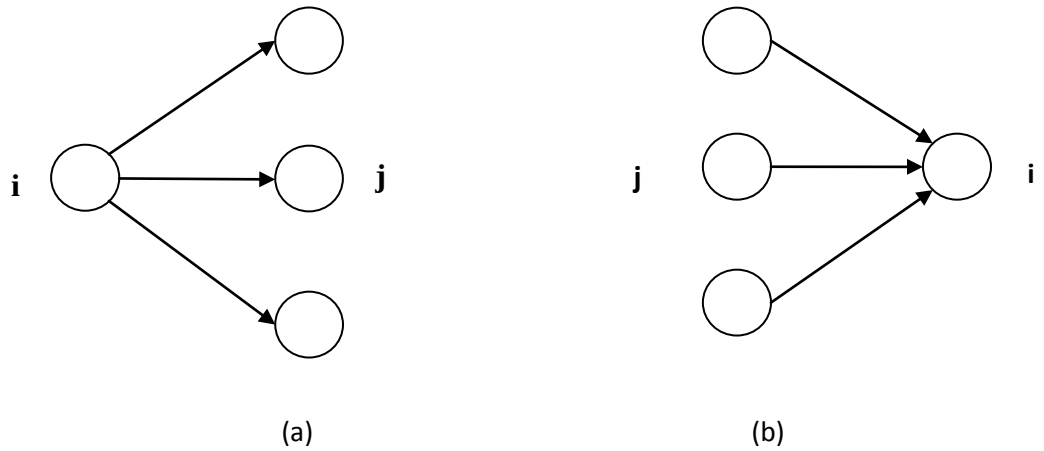


Figure 16: (a) Hub webpage pi has many out-bound hyperlinks. (b) Authority webpage pi has many in-bound hyperlinks.

Authority weight of a page is calculated as a sum of hub weights pointing to it and weight of a hub as a sum of weights of authorities pointed to by it. Let S be a set of pages for which hub and authority weights are being calculated, n is number of pages in the set. Then H is a subset of S containing pages acting as hubs and A is a subset of S containing authorities. Since each page can be an authority and a hub, A and H overlap. For every page i in its hub role F (i) is the number of outgoing links. For every page i in its authority role B (i) is the number of incoming links. The n-dimensional vector of authority weights is denoted as a, and vector of hub weight as h. Then hub and authority weight are calculated by the formula

$$a(i) = \sum_{j \in B(i)} h(j)$$

$$h(i) = \sum_{i \in F(j)} a(j)$$

In matrix-vector terms

$$a = W^T \mathrm{h} \qquad \text{and} \qquad \mathrm{h} = \mathrm{Wa}$$

This process for computing the hub and authority weights is iterative in nature. Initially all authority and hub weights are set to 1. At each iteration the operations O (Out) and I (In) are performed. The O operation updates the authority weights and the I operation updates the hub weights both using the above equation. A normalization step is then applied, so that the vectors a

and h become unit vectors in some norm. The algorithm iterates until the vector converges. This idea was later implemented as the HITS (Hyperlink Induced Topic Distillation) algorithm [17]. The HITS algorithm is now a part of the CLEVER searching project of the IBM Almaden Research Center[17]. The algorithm is summarized in Figure 17.

HITS
Initialize all weights to 1
Repeat until the weights converge:
For every hub i H

$$h(i) = \sum_{i \in F(j)} a(j)$$

For every authority I A

$$a(i) = \sum_{j \in B(i)} h(j)$$

Normalize

Figure 17: HITS algorithm [7]

So in the HITS algorithm, the first step is to retrieve the set of results to the search query. The computation is performed only on this result set not across all Web pages. The algorithm performs a series of iterations each consisting of two basic steps:

- **Authority Update**: It update each node's authority score to be equal to the sum of the Hub scores of each node that points to it. That is, a node is given a high authority score by being linked to by pages that are recognized as Hubs for information [19].
- **Hub Update**: It Updates each node's hub Score to be equal to the sum of the authority scores of each node that it points to. That is, a node is given a high hub score by linking to nodes that are considered to be authorities on the subject [19].

HITS and PageRank both are iterative algorithm and both of these algorithms are based on based on linkage of the documents on the web. But there are some major differences which distinguish HITS from PageRank these differences are as follows:

- HITS algorithm is executed at query time and PageRank algorithm is executed at indexing time with the associated hit on performance that accompanies query time processing. Thus the hub and authority scores assigned to a page are query-specific.
- The PageRank algorithm is used by Google search engine and because of PageRank algorithm Google is the best search engine but HITS algorithm is not commonly used by search engines.
- HITS algorithm computes two scores per document, hub and authority as opposed to PageRank algorithm which compute a single score which is PageRank value of the web page.
- HITS algorithm is processed on a small subset of relevant documents not all documents as is the case with PageRank.

2.7.3. SALSA (Stochastic Approach for Link Structure Analysis) Algorithm

An alternative algorithm, SALSA was proposed by Lempel and Moran [33]. It combines ideas from both HITS and PageRank. This approach is based upon the theory of Markov chains and relies on the stochastic properties of random walk. It takes a result set R as input, and constructs a neighborhood graph from R in precisely the same way as HITS. Similarly, it computes an authority and a hub score for each vertex in the neighborhood graph and these scores can be viewed as the principal eigenvectors of two matrices. Like Kleinberg's algorithm, SALSA starts with a similarly constructed base set. It then performs a random walk by alternately

(a) Going uniformly to one of the pages which link to the current page.

(b) Going uniformly to one of the pages linked to by the current page.

The authority weights are defined to be the stationary distribution of the two-step chain doing first step (a) and then (b), while the hub weights are defined to be the stationary distribution of the two-step chain doing first step (b) and then (a). However, instead of using the straight adjacency matrix that HITS uses, SALSA weighs the entries according to their in and out-degrees. As the ca HITS visualizes the graph G as a bipartite graph, where hubs point to authorities, the SALSA algorithm performs a random walk on the bipartite hubs and authorities graph alternating between

the hub and authority sides. The random walk starts from some authority node selected uniformly at random. The random walk then proceeds by alternating between backward and forward steps. When at a node on the authority side of the bipartite graph, the algorithm selects one of the incoming links uniformly at random and moves to a hub node on the hub side [7]. When at node on the hub side the algorithm selects one of the outgoing links uniformly at random and moves to an authority. The authority weights are defined to be the stationary distribution of this random walk.

$$Pa_{(i,j)} = \sum_{k:\, kB(i)\setminus B(j)} \frac{1}{|B(i)|} \frac{1}{|F(k)|}$$

Ga = (A, Ea) denotes the authority graph, where there is an (undirected) edge between two authorities if they share a hub. This Markov Chain corresponds to a random walk on the authority graph Ga, where anyone move from authority i to authority j with probability Pa (i,j) . Let Wr denotes the matrix derived from matrix W by normalizing the entries such that, for each row the sum of the entries is 1 and let Wc denote the matrix derived from matrix W by normalizing the entries such that, for each column the sum of the entries is 1. Then the stationary distribution of the SALSA algorithm is the principal left eigenvector of the matrix $M_S = W_c{}^T W_r$.

If the underlying authority graph Ga consists of more than one component, then the SALSA algorithm selects a starting point uniformly at random, and performs a random walk within the connected component that contains that node. Let j be a component that contains node i, let Aj denote the set of authorities in the component j, and Ej the set of links in component j. Then the weight of authority i in component j is

$$a_i = \frac{|Aj|}{|A|} \frac{|B(i)|}{|Ej|}$$

If the graph Ga consists of a single component (Such graphs are referred as authority connected graphs) that is the underlying Markov Chain is irreducible, then the algorithm reduces to the INDEGREE algorithm [7]. Furthermore, even when the graph Ga is not connected if the starting point of the random walk is selected with probability proportional to the popularity (in-degree) of

the node in the graph G, then the algorithm again reduces to the INDEGREE algorithm. This algorithm is referred to as PSALSA which is popularity-SALSA by Borodin [6].

The SALSA algorithm can be thought of as a variation of the HITS algorithm. In the operation of the HITS algorithm the hubs broadcast their weights to the authorities and the authorities sum up the weight of the hubs that point to them. The SALSA algorithm modifies the I (In) operation so that instead of broadcasting each hub divides its weight equally among the authorities to which it points. Similarly, the SALSA algorithm modifies the O (Out) operation so that each authority divides its weight equally among the hubs that point to it. Therefore,

$$a_i = \sum_{j:\, j \in B(i)} \frac{1}{|F(j)|} h(j) \qquad \text{And} \qquad h_i = \sum_{j:\, j \in F(i)} \frac{1}{|B(j)|} a(j)$$

2.8 Dangling Pages

Dangling pages are pages which do not have any out link and does not provide reference to other pages. An important aspect of outbound links is the lack of out links on web pages. When a web page has no outbound links, its PageRank cannot be distributed to other pages. These Dangling pages create many issues like philosophical, computational and storage issue. Dangling pages are also called Hanging pages and Zero-out link pages [48]. Such hanging pages can act as sinks or black holes for PageRank a key factor in the Google search algorithm. The Web was mostly a static when the original PageRank algorithm was developed so it was easy to crawl the entire Web also there were not many changes in the link structure of the web.But now the web is evolving into a dynamic one driven by databases. The dynamic behavior of web causes the Dangling pages to create Link rot problem [14]. The link rot is a problem in which the links which were working at one time is not working now a day. The reason behind is that the content are removed from that link or URL of that link is changed or the links are broken and sometimes it will return HTTP code 403 or 404 [43]. The pages which return this HTTP code are called penalty pages. There are several algorithms push-back, self-loop, jump-weighting and BHITS proposed by N. Eiron [14] to adjust the ranks of pages with links to penalty pages. When a user reaches at any dangling node then there is no other option to move further from that node.

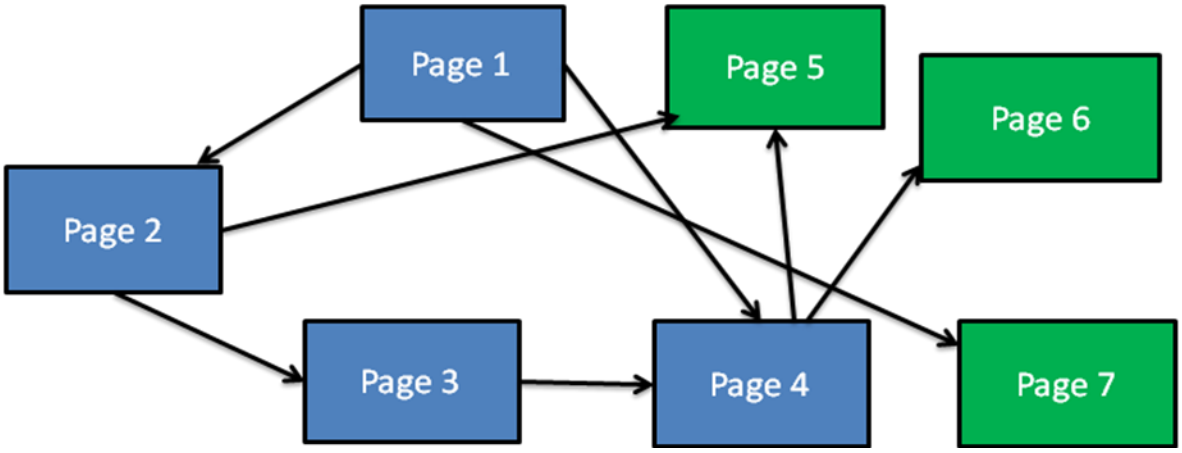


Figure 18: A Directed Web Graph with three Dangling Web Pages (Green node)

In the figure 18 the node 5 and node 6 represent the dangling pages because they do not have out links to any other pages when a user reaches at page 5 or page 6 then they do not have any option to move from there. Node 1 to node 4 are non-dangling nodes (blue color nodes) because each of them have outlinks to any other pages in the web graph.

The BHITS algorithm resembles the HITS algorithm [11] is also used for handling dangling pages. The numbers of dangling pages are increased day by day because of the evolving nature of the web. Since the number of dangling nodes is growing so rapidly there is need to find out some solution for this [27]. Removal of dangling pages from the web graph is not the solution for handling dangling web pages because removal of these pages also create new problem.

The hyperlink structure of the web defines as a directed graph which can be expressed as a sparse matrix by applying the equation 4 on the web graph. The sum of nonzero rows of matrix M is equal to 1. The matrix M contains a 0^T row corresponding to each dangling node. Brin and Page define the PageRank vector, a vector holding the global measure of importance for each page, to be the stationary vector for a Markov chain related to P [9, 4].This definition is intuitive as the stationary vector gives the long run proportion of time the chain will spend in each state. One problem with the hyperlink matrix M created strictly from the web's structure is that it is not

stochastic when the dangling node is present there and a Markov chain is defined only for stochastic matrices. PageRank of the dangling nodes depends strongly on that of the non-dangling nodes but not vice versa.

2.8.1 Issues Related to Dangling Page

In order to compute the page rank vector the transition probability matrix need to be stochastic matrix. A matrix is stochastic if the sum of all the row in the matrix is equal to 1 or greater than zero. But when there is dangling node in the graph then the row corresponding to dangling node contains all zero O^T. This O^T row of the matrix makes the matrix non stochastic as sum of the row is not 1.For example considers the web graph in Figure 19:

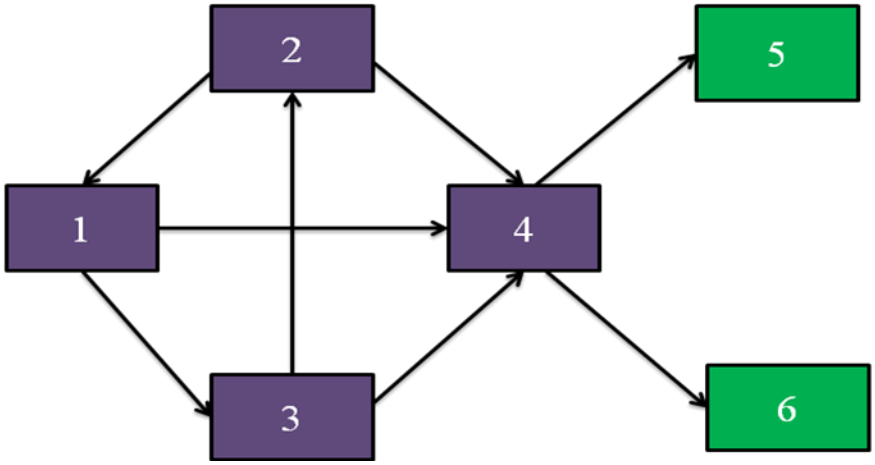


Figure 19: A directed Web Graph with 2 Dangling Pages (Green node)

According to equation 4 the transition probability matrix of web graph presented in figure 19 is

	1	2	3	4	5	6
1	0	0	1/2	1/2	0	0
2	1/2	0	0	1/2	0	0
3	0	1/2	0	1/2	0	0
4	0	0	0	0	1/2	1/2
5	0	0	0	0	0	0
6	0	0	0	0	0	0

Figure 20: Transition Probability Matrix of Web Graph having 2 Dangling Nodes

In the above transition probability matrix M row corresponding to dangling node 5 and 6 contain 0^T row so the above matrix is not stochastic.

There are two main issues which related to dangling pages. First is philosophical and second is computational issue. The philosophical issue occurs due to excluding the dangling node from the page rank computation because Surgey Brin and Larry Page remove the entire dangling node from the graph during the computation in the original page rank algorithm [10]. After that when the page rank of non-dangling nodes is calculated the dangling pages are reinserted back to graph without disturbing the result of the non-dangling pages. Some dangling nodes should receive high page rank. For example, a very authoritative pdf file could have many inlinks from respected sources and thus should receive a high page rank. Removing the dangling nodes biases the PageRank vector unjustly and it also create more other problem [31]. But removing the dangling node from the graph is not the solution because sometimes removal of dangling node creates more dangling node. The removal of dangling node also produce the inaccurate page rank of non-dangling node because removal of dangling pages also effect the out degree of non-dangling pages which has a link to these dangling pages. Here is the example which explains the above problem associated with removal of dangling pages.

From figure 21 it can be seen that removing one dangling page can cause new dangling pages and hence, removal of dangling pages can be an iterative process. After the PageRank calculation is finished, PageRank can be assigned to the formerly removed pages based on the PageRank algorithm. In the figure 21(i) page 5 (green node) is dangling node if dangling page 5 is removed from the graph (see in figure 21(ii)) it create new dangling node which is page 4 in the graph as in figure 21(iii) and again the removal of web page 4 (figure 21 (iv)) also create new dangling node which is node 3 (figure 21 (v)) and removal of dangling node 3 is shown in figure 21 (vi) so the removal of dangling node becomes iterative process until all the dangling nodes are remove from the graph. So removing the dangling node sometimes makes loss of great amount of useful data from the web.

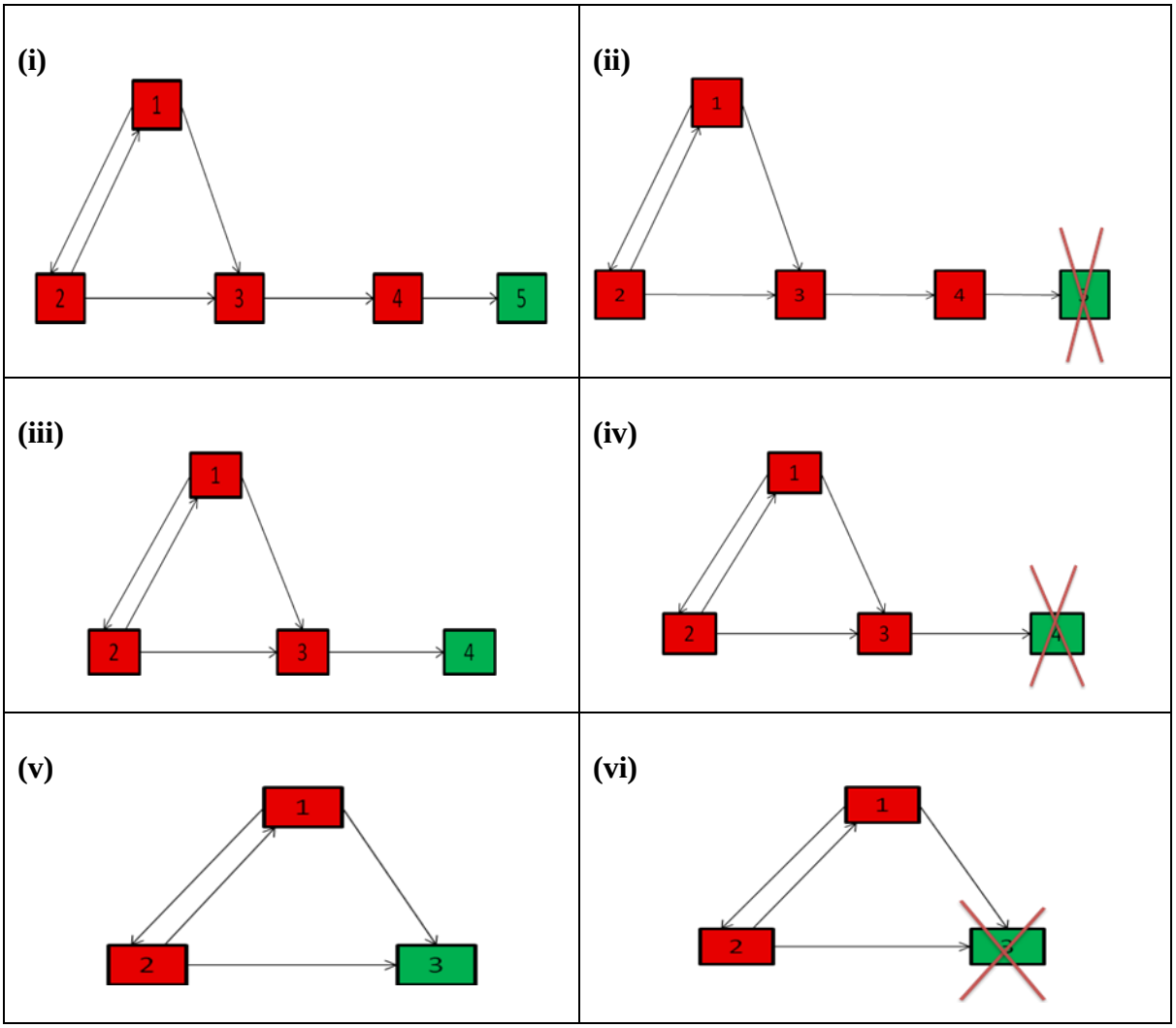


Figure 21: Iterative Procedure of Removing the Dangling Node from the Web Graph

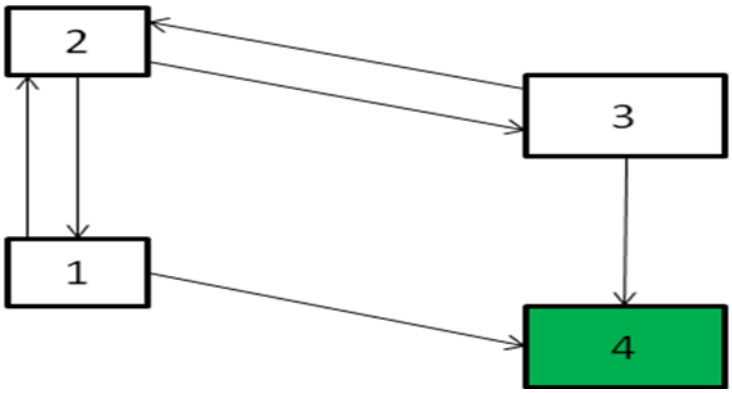


Figure 22: A Directed web graph with One Dangling Page (Green node)

In the above figure 22 if dangling page 4 is removed from the graph then it also affect the out degree of page 1 and page 3 and in this way removal of dangling node produce inaccurate value of non-dangling nodes because in the PageRank calculation the out degree of each page which provide inlinks to computing page is also considered.

Computational issue is also associated with the dangling nodes because the dangling nodes are excluded from most of the computations the operation count depends to a large extent only on the number of non-dangling nodes as opposed to the total number of web pages. The algorithm is much faster because it operates on a much smaller matrix so it convergence in less number of iteration. The efficiency of the algorithm increases as the number of dangling nodes increases by the dangling nodes [20]. But in the page rank computation the dangling node also effect the page rank value of non-dangling node as it only absorb the value, the dangling node never distribute their page rank value to other node because they do not have any out link to other page. So dangling node increase the computation issue. As the removal of dangling node also affect the out degree of non-dangling page so indirectly removal of dangling pages affect the page rank value of non-dangling pages. There is need to handle or decrease these philosophical and computational issue associated with dangling pages. Some approaches are developed for handling the dangling node in page rank computation, these approaches are discussed in next section.

2.8.2 Approaches to Handle Dangling Node

1) Surgey Brin and Larry Page suggest that removes the entire dangling node from the web graph during page rank computation. After that when the page rank of non-Dangling nodes is calculated the dangling pages are reinserted back to graph without disturbing the result of the non-dangling pages. Simply removing the dangling nodes biases the Page Rank vector unjustly and additionally it also create more other problem. In [23] is suggested to add the nodes for the final few iterations of the computations. But addition of dangling node in last few iteration does

not produces accurate page rank value of dangling node. This approach also produce philosophical problem because the dangling node is excluded from the page rank computation.

2) The another approach to deal with dangling node given by Surgey Brin and Larry Page suggest that replace the each 0^T row of the transition probability matrix with a dense vector. The main solution for this problem is to use a uniform vector e^T/n (e is the vector of all ones) [26] but with this solution problem of the storage requirement increase considerably. The number of dangling pages exceeds the number of non-dangling pages [14]. This case is represented on the directed web graph by having a directed edge from a dangling node to all web pages in the graph including itself. For example after applying this solution the web graph presented in figure 19 the figure 19 is changed to figure 23. This is implemented in the transition probability matrix by adding a non-zero row to matrix in place of the row representing the dangling node .In figure 20 the transition probability matrix it can be seen that the row corresponding to node 5 and node 6 contain all zero after applying the above approach the transition probability matrix is modified into the transition probability matrix presented in Figure 24.

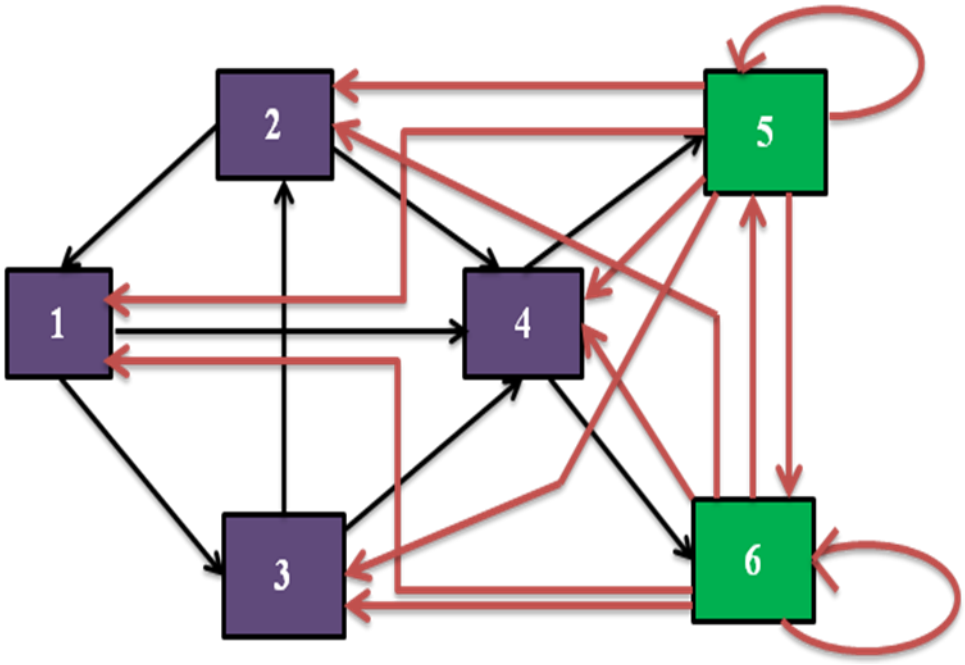


Figure 23: Web Graph after connecting Dangling node to the each Node in the Web Graph including Itself

In the figure 23 the dark red color links shows the links which are created from the dangling node (Green node) to all non-dangling node. In the figure 24 the row corresponding to node 5 and 6 is now changed and sum of row corresponding to dangling node 5 and node 6 are equal to 1 and so the matrix become stochastic. On this transition probability matrix now markov chain can be applied to compute the page rank vector.

	1	2	3	4	5	6
1	0	0	1/2	1/2	0	0
2	1/2	0	0	1/2	0	0
3	0	1/2	0	1/2	0	0
4	0	0	0	0	1/2	1/2
5	**1/6**	**1/6**	**1/6**	**1/6**	**1/6**	**1/6**
6	**1/6**	**1/6**	**1/6**	**1/6**	**1/6**	**1/6**

Figure 24: Transition Probability Matrix after Replacing the Row corresponding to Dangling Nodes with e^T/N

3) The rows corresponding to dangling nodes would be zero in the matrix M and cause philosophical problem if not considered into the PageRank algorithm. The other approach to deal with this zero rows and force the matrix to be stochastic is that, lumps all dangling nodes into a single node [30]. For this a hypothetical node is added to the web graph and the entire dangling node in the web graph is connected to hypothetical node and a self loop from the hypothetical node which point back to itself is constructed [20]. By creating this hypothetical node in the graph make the matrix becomes stochastic which is necessary for computing the page rank vector as the Markov chain is only defined for stochastic matrix. Now the entire dangling nodes in the graph have at least 1 outlink and target of all these out-link is hypothetical node. The

hypothetical node also has 1 out degree because of self loop. From this solution, philosophical issue is solved because now the dangling node is not excluded from the graph before the calculation. Page rank of dangling node is calculated along with the non-dangling node. The figure 25 graphically the result of applying the approach defined above on figure 19.

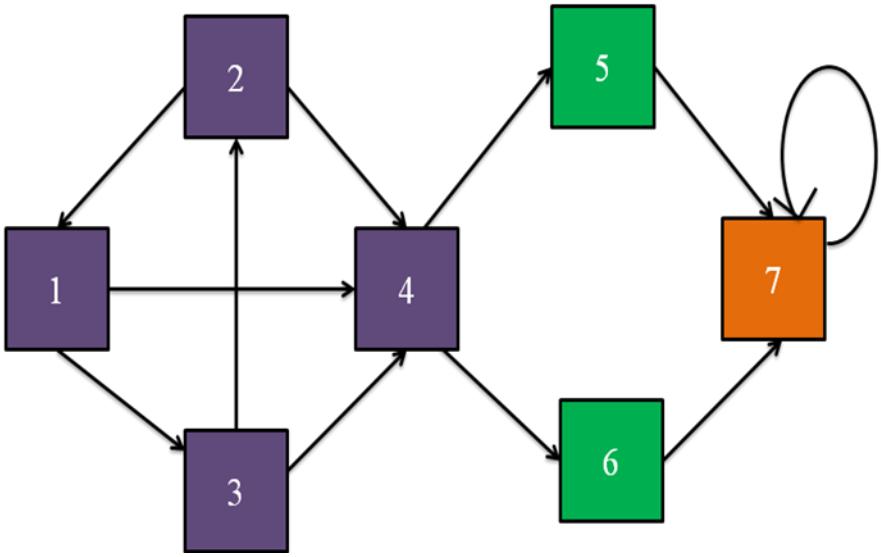


Figure 25: A Directed Web Graph with after Lumping all Dangling Node into Hypothetical Node (Orange Node)

In the figure 19 the node 5 and 6 are dangling node but after adding hypothetical node (orange color node) to figure 19 and connecting the dangling node (green nodes) to hypothetical node 7 and also constructing a self loop on the node 7 , then it converted into figure 25. Now in the web graph no nodes have the 0 out degree. The transition probability matrix of above web graph is

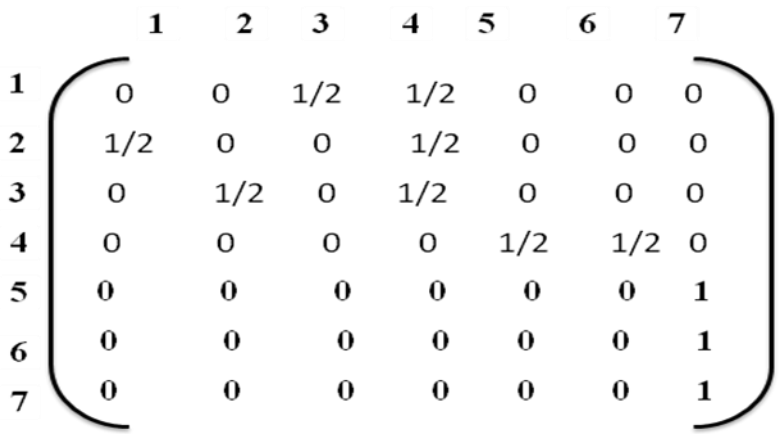


Figure 26: Transition Probability Matrix after Lumping all the Dangling Node into a Hypothetical node

From the figure 26 it can be seen that the row corresponding to dangling node now not contaion all 0, because the now the node 5 and node 6 has 1 out degree, they both have outlink to hypothetical node 7. The hypothetical node also has a self loop so row corresponding to hypothetical node is also contain 1 out degree. The matrix is now stochastic as no rows in the matrix contain all zero. Now markov chain can be apply to compute the page rank vector. By implementing this method page rank value of all the nodes non-dangling, dangling or hypothetical are obtained. The value of hypothetical node is always having a high Page Rank and it can be ignored as defined in [20].

Chapter 3

Problem Statement

PageRank algorithm is link analysis algorithm used by the Google Search engine. This algorithm assigns a numerical value to each page in the World Wide Web so that it measures the relevant importance of each page rank with other pages on the web. On the web there are some web pages called dangling pages which do not have any out link or reference to other pages. Dangling pages are also called Hanging pages or Zero-out link pages. When a web page has no outbound links then its PageRank cannot be distributed to other pages. These dangling pages create philosophical and computational issues. In the existing PageRanK algorithm, the dangling pages are removed from the web graph and then PageRank is calculated for the non-dangling pages. Removal of dangling node cause philosophical issue, because the exclusion of dangling node from the PageRank computation. The removal of dangling node is not a solution for handling dangling node because sometimes removal of dangling node creates more dangling nodes in the web graph. Some dangling nodes may get high PageRank and the removal of those nodes is not justified. For example, a quality pdf file may have many inlinks from good sources and it should receive a high rank, but in the original PageRank computation dangling nodes are removed. Dangling pages also affect the PageRank value of non-Dangling node. The removal of dangling node may be an iterative process as discussed in Literature survey. The matrix M is also not stochastic when the dangling nodes are presented in the web graph, because the rows corresponding to dangling node contain all zero.

In the previous chapter some exiting approach to handle dangling node is discussed. The approach discussed by the Surgey Brin and Larry Page that remove the each dangling node from the graph during the page rank computation produces philosophical issue. The other approach to handle dangling node is to replace the row corresponding to dangling pages in the transition probability matrix with a dense vector. The main solution for this approach is to use a uniform vector e^T/n (e is the vector of all ones) but with this approach problem of the storage requirement increase considerably. The approach to deal with the zero rows and forcing the matrix to be stochastic is to connect a hypothetical node to the web graph and connecting each dangling node

in the web graph to hypothetical node and constructing the Self loop from the hypothetical node which point back to itself. This hypothetical node solution solves the problems related to philosophical issue by including the dangling node in the PageRank computation. But the computational problem is increased with this solution. The number of iterations also increased with this approach. Because of hypothetical node the PageRank algorithm take lagre number of iteration to converge the algorithm.

To resolve all these issues related to dangling node a algorithm named **"A Modified Algorithm to Handle Dangling Pages using Hypothetical node in PageRank Computation"** has been proposed. The proposed algorithm uses the concept of hypothetical node and gives better performance without increasing computational cost. With the help of this algorithm philosophical issue has been resolved by including the dangling node in the PageRank computation. Matrix has become stochastic although dangling node is present in the web graph. Computational problem means the numbers of iterations are reduced which was increased due to addition of hypothetical node. By solving the computational problem and all the issue related to dangling node the page rank value of the each node (Non-dangling and dangling node) presented in the Web graph is computed in less number of iteration.

Chapter 4

Proposed Modified Algorithm

4.1 Introduction

For solving the issues related to dangling node with the concept of hypothetical node without increasing the computational problem a modified algorithm is proposed. The computational problem which is generated due to addition of hypothetical node is solved by not comparing the value of hypothetical node to become unchangeable in the iterations for the algorithm to converge. The algorithm is going to be converged when value of all other node except the hypothetical node is same in both $(i-1)^{th}$ iteration and i^{th} iteration. At that point accurate value of dangling node is obtained because when value of all non-dangling node is set or become unchangeable then value of dangling node at that point also become unchangeable because the inlinks to these dangling pages are always from these non-dangling pages. According to the formula described in equation 3 if there is no change in the page rank value of all the pages which provide inlink to page for which the page rank value is calculated than the page rank value of computing page is always same untill the page rank value of any inlink page is not changed.

In the existing PageRank algorithm, from which the dangling node is excluded from computation there accurate page rank value of dangling node is not calculated. The previous approach to handle dangling pages using hypothetical node solves the philosophical issue by including all the nodes in the computation. But because of this hypothetical node the number of iteration is increased too much. The Page rank value of hypothetical node is always too high as compare to all other node so it can be ignored. If for converge the algorithm the value of hypothetical node is not compared then number of iteration can be decreased. The value of all other nodes is set in few iteration only hypothetical node take large number of iteration to set their value. The existing page rank algorithm compare the value of all the nodes presented in the web graph to converge the algorithm .For solving the above problem modified algorithm is proposed which does not compare the value of hypothetical node for the algorithm to converge and produce accurate page rank value of dangling node along with non-dangling node.

4.2 Proposed Algorithm

Input – Graph of web Pages

Output- Page Rank of every web page

1. K ← 1, d←0.85.
2. Repeat for i from 0 to no_of_Vertexes
    a. Prin ←0.
    b. Repeat
        for m from 0 to no_of_Vertexes
            If (adjacency matrix[m][i]==1) then
            Prin = Prin + Vertexes[m].Pr/Vertexes[m].out
        End if.
    c. End loop.
3. Vertexes[i].Prin = (1-d) + (d* Prin).
4. End loop.
5. Repeat
    For l from 0 to no_of_Vertexes
    a. Compare pr of each Vertex with respective Prin of each Vertex leaving Hypothetical Vertexes
6. If same value are not achieved then
    a. Repeat
        For h from 0 to no_of_Vertexes
        (i) Vertexes[h].Pr = Vertexes[h].Prin
        (ii) End loop.
    b. Repeat from step 2
    c. K++.
    End if
7. End loop.

In the modified algorithm first the value of damping factor d and value of k is set. The value of d is set equal to 0.85 which is commonly used value for d. K is used to show the number of

iteration. Out represent the out degree of node. After that the intermediate page rank value (Prin) of each node is set equal to 0. Again run the for loop is run for all the vertex and wherever there is one in the matrix corresponding to some node then calculate the intermediate page rank value of node by the use of formula Prin = Prin + vertexes[m].pr/vertexes[m].out. Then the page rank value of the each node is calculated at the end of first for loop by the use of formula Vertexes[i].Prin = (1-d) + (d* Prin). Again a for loop is constructed for all the vertexes, for comparing the page rank value of each node (prin) at any iteration with the previous page rank value (pr) leaving the value of hypothetical vertexes. If the same results are not achieved then again for loop is run for all the vertexes and store the page rank value of all the vertexes in pr by Vertexes[h].Pr = Vertexes[h].Prin. Again repeat from step 2 and increment the value of K. Else if the same results are achieved then the algorithm is converged.

On this modified algorithm if any web graph is run then it takes less number of iteration than the existing PageRank algorithm. Previously the algorithm is converged when at some iteration all the nodes in the graph is same value in the previous iteration. But now the algorithm is waiting until value of all nodes except the hypothetical node to become unchangeable in iterations. By this modified algorithm the computational issue is solved along with philosophical issue with hypothetical node approach. Now the page rank value of all the node (non-dangling and dangling) can be calculated without increasing the computational cost. Because of the high value of hypothetical node it can be ignored. The hypothetical node is added for solving the dangling problem and it is not a real page hence its accurate page rank value does not matter. At the point when Page rank value of each node except hypothetical node is set then at that point some value of hypothetical node is also obtained. The value of hypothetical node at this point may be inaccurate but it does not affect the value of dangling and non-dangling node so it does not matter whether the value of hypothetical node is accurate or inaccurate. Now the dangling node is handled with this modified algorithm. The matrix also becomes stochastic and it does not have 0^T row in the transition probability matrix corresponding to dangling page. The philosophical issue and computational issue related to dangling node is also solved. Now page rank value of the all the nodes in the graph is obtained result in less number of iteration than the number of iteration it takes when the web graph was run on the existing PageRank algorithm.

Chapter 5
Experimental Results

5.1 Introduction

In this chapter the experiment result are shown by applying the hypothetical node approach on any web graph and then running the existing and modified algorithm on that graph and at last comparing the number of iteration that each algorithm take to converge the algorithm. It is also describes how the number of iteration is increased due to addition of hypothetical node or how the computational problem is increased. After that how the modified algorithm solves the computational problem is shown by running the web graph on the modified algorithm. The modified algorithm is implemented in C++. The calculation is performed up to 10 decimal places. The number of vertexes, the number of outgoing links from each vertex and the target node of each outgoing links is entered as an input. The program produces as output the Page Rank value of each node after the convergence of iteration and it also shows the out degree of each node. The output of the program is stored in a file to analyze the intermediate page rank value of each page. Two experiments are performed to show that the modified algorithm is working with any web graph. In each experiment there are two cases

Case 1: Run the web graph on the existing algorithm.

Case 2: Run the web graph on the proposed modified algorithm.

5.2 Experimental Results

5.2.1 Result of First Experiment

First the web graph shown in figure 25 is run on the existing page rank algorithm. Figure 25 is result of applying the hypothetical node approach on the web graph shown in Figure 19. After that the web graph is run on the modified algorithm which is case 2. The web graph in figure 26 contain the 7 node in which there are two dangling node which is node 5 and node 6 (green node) and a hypothetical node which is node 7 (dark orange color node). After running the web graph on both the algorithm the number of iteration is compared that each existing and proposed modified algorithm takes for converged.

Case 1: Run the above Web graph on Existing Algorithm

When the existing PageRank algorithm run on web graphs shown in figure 25 then, it takes 135 iterations to converges the algorithm. The result after applying the existing algorithm is shown in figure 27.

1:	0.1499999765	0.1499999765	0.1499999765	0.1499999765	0.1499999765	0.1499999765	0.1499999765
2:	0.1924999702	0.2987499563	0.2349999646	0.2349999646	0.2349999646	0.2137499681	0.5324999271
3:	0.2346457995	0.3709999494	0.2711249612	0.2891874595	0.3012291250	0.2498749640	0.9840623805
4:	0.2551166314	0.4136475157	0.2984194042	0.3215996098	0.3370530809	0.2652280879	1.4548915117
5:	0.2672001092	0.4402311662	0.3134029156	0.3394831568	0.3583200007	0.2768282271	1.8985968032
.	.....						
.	.....						
.	.....						
.	.....						
36:	0.2850075278	0.4764972300	0.3343840189	0.3657596627	0.3886394354	0.2921131876	4.8374533588
37:	0.2850075285	0.4764972307	0.3343840189	0.3657596634	0.3886394361	0.2921131883	4.8404751923
38:	0.2850075285	0.4764972307	0.3343840189	0.3657596634	0.3886394361	0.2921131883	4.8452270279
39:	0.2850075285	0.4764972307	0.3343840189	0.3657596634	0.3886394361	0.2921131883	4.8430437519
40:	0.2850075285	0.4764972307	0.3343840189	0.3657596634	0.3886394361	0.2921131883	4.8470828121
41:	0.2850075285	0.4764972307	0.3343840189	0.3657596634	0.3886394361	0.2921131883	4.8486602289
.	.....						.....
.	.....						.....
.	.....						.....
.	.....						.....
131:	0.2850075285	0.4764972307	0.3343840189	0.3657596634	0.3886394361	0.2921131883	4.8575989221
132:	0.2850075285	0.4764972307	0.3343840189	0.3657596634	0.3886394361	0.2921131883	4.8575989228
133:	0.2850075285	0.4764972307	0.3343840189	0.3657596634	0.3886394361	0.2921131883	4.8575989235
134:	0.2850075285	0.4764972307	0.3343840189	0.3657596634	0.3886394361	0.2921131883	4.8575989242
135:	0.2850075285	0.4764972307	0.3343840189	0.3657596634	0.3886394361	0.2921131883	4.8575989242

Figure 27: Result after Applying the Existing PageRank Algorithm on Web Graph shown in figure 25.

The last two rows (value in red color and last two value in yellow color) in figure 27 shows the page rank value of the each node at 134th and 135th iteration it is clear from the figure that value

of each node is same at both the iteration so algorithm is converged here. The value in green color shows the number of iteration. Hypothetical node is always the last node in the web graph because it is added at last in the web graph if the dangling nodes are present in the web graph. From the above result it is also observed that value of all other nodes except hypothetical node is same at 37th and 38th iteration (value in light purple color) but the value of hypothetical node at 37th and 38th iteration is not same. Because of hypothetical node iteration is went up to 135 iterations. The value of hypothetical node is always very high in comparison to other nodes. At the 135th the iteration the value of hypothetical node is 4.8575989242 which is very high when compared it to all other value. The value of hypothetical node can be ignored.

Case 2: Run the Web Graph on the Proposed Modified Algorithm

Figure 28 shows the result of applying the web graph (shown in figure 25) on the modified algorithm. Now the number of iteration is 38. From the fifure 28 it can be observed that the algorithm is now converged at the 38th iteration when the value of all other nodes except hypothetical node is same. From the figure it is clear that the value of all the node except hyothteical node is same in both the iteration at the 37th and 38th (value in red color) but the value of hypothetical node is different. The value in green color show the numer of iteration.

1:	0.1499999765	0.1499999765	0.1499999765	0.1499999765	0.1499999765	0.1499999765	0.1499999765
2:	0.1924999702	0.2987499563	0.2349999646	0.2349999646	0.2349999646	0.2137499681	0.5324999271
3:	0.2346457995	0.3709999494	0.2711249612	0.2891874595	0.3012291250	0.2498749640	0.9840623805
4:	0.2551166314	0.4136475157	0.2984194042	0.3215996098	0.3370530809	0.2652280879	1.4548915117
5:	0.2672001092	0.4402311662	0.3134029156	0.3394831568	0.3583200007	0.2768282271	1.8985968032
.	.....						
.	.....						
.	.....						
.	.....						
.	.....						
35:	0.2850075271	0.4764972279	0.3343840167	0.3657596627	0.3886394332	0.2921131869	4.8338982634
36:	0.2850075278	0.4764972307	0.3343840189	0.3657596627	0.3886394354	0.2921131876	4.8374533588
37:	0.2850075285	0.4764972307	0.3343840189	0.3657596634	0.3886394361	0.2921131883	4.8404751923
38:	0.2850075285	0.4764972307	0.3343840189	0.3657596634	0.3886394361	0.2921131883	4.8430437519

Figure 28: Result after applying A Modified Algorithm on Web Graph shown in figure 25

The value of hypothetical node at 37^{th} iteration is 4.8404751923 and value of hypothectical node at 38^{th} iteration is 4.8430437519 both the value are not same but the algorithm is converged according to modified algorithm. The value of hypothetical node is still high and can be ignored because of having high value when compared to page rank value of other node in the graph. The iteration at both the case is shown in table 2. From the table it is clear that number of iteration in case 2 is less than the number of iteration in case 1.

Table 2: Number of Iterations in both the Cases in first experiment

Case	**Number of iteration**
Case 1	135
Case 2	38

5.2.2 Result of Second Experiment

The second experiment is performed on web graph shown below in Figure 29. In this web graph there are 7 node where node 2 and node 7 (green node) is dangling node because they do not have any outlink and purple color nodes are non-dangling node. First both the existing and modified algorithm run on this web graph then the number of iterations that each algorithm take is compare at last to show that modified algorithm decrease the number of iterations to converge the algorithm.

After applying the Hypothetical node solution the web graph (shown in figure 29), it is converted into the Figure 30. In figure 30 the Hypothetical node, node8 (orange color node) is added to above web graph shown then connect dangling node which is the node 2 and node 7 to the hypothetical node 8 and construct a self loop on node 8 is constructed. By adding the hypothetical node philosophical issue related to dangling pages is solved but computational

problem is increased, for reducing the computational problem proposed modified algorithm is used for computing the page rank value of each node instead of existing PageRank algorithm.

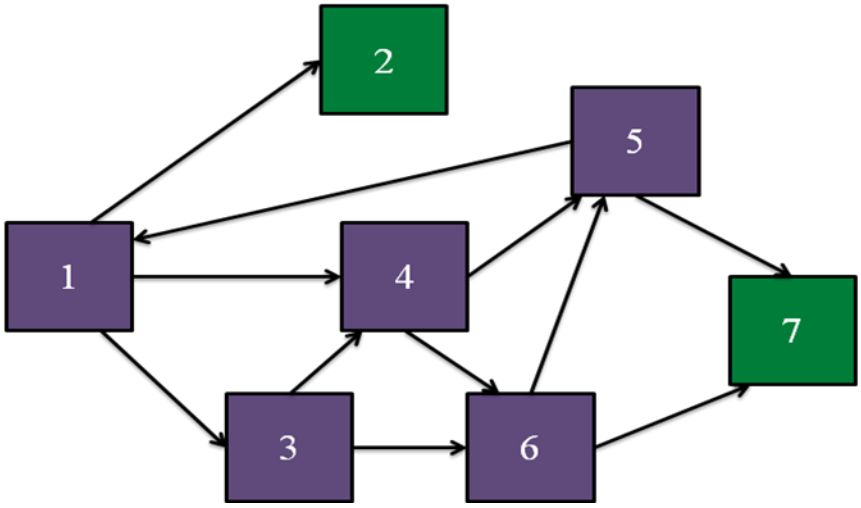


Figure 29: A Directed Web Graph with Two Dangling Node (Green Node)

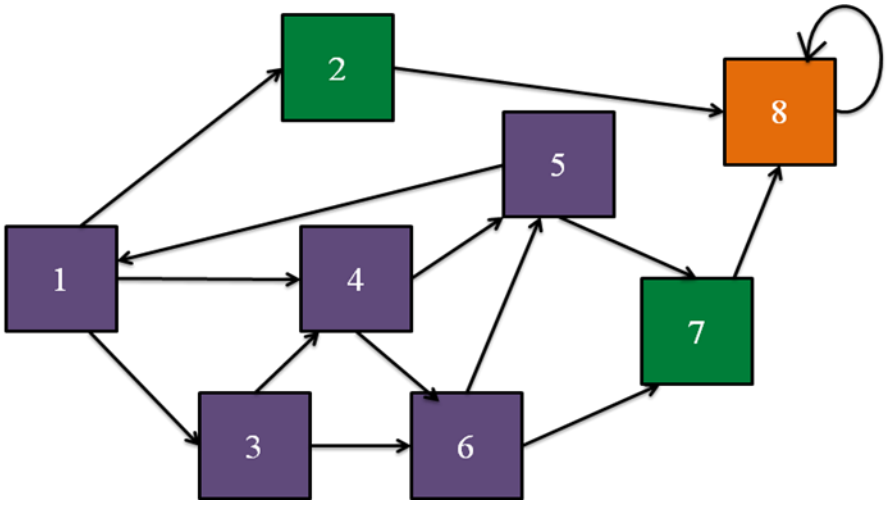


Figure 30: Web Graph after Connecting Dangling Nodes to Hypothetical Node (Orange node)

Case 1: Run the Web Graph on Existing PageRank Algorithm

Now on the above web graph the existing algorithm is run. The result of applying the existing algorithm on the web graph is shown in figure 31. From the figure 31, it can be seen that the

algorithm is converged at 135th iteration. The value in the green color shows the number of iteration. The algorithm is converged at the 135th iteration because the page rank value of all the nodes is same in the both 134th and 135th iteration (value in red color and last two value in yellow color).The value of all the node expect the hypothetical node is same at the 67th and 68th iteration but hypothetical node value (values in yellow color) is still changing from 68th iteration to 134th iteration. Because of the hypothetical node value the algorithm goes up to 135 iteration. The result after applying the existing algorithm on the web graph (shown in figure 30) is

1:	0.1499999765	0.1499999765	0.1499999765	0.1499999765	0.1499999765	0.1499999765	0.1499999765	0.1499999765
2:	0.1924999702	0.1924999702	0.3199999542	0.3199999542	0.2774999598	0.1924999702	0.1924999702	0.5324999271
3:	0.2286249667	0.2045416357	0.3681666149	0.4765416045	0.4219999446	0.2286249667	0.2286249667	0.9298748856
4:	0.2695666306	0.2147770517	0.4091082788	0.5277186830	0.5550603515	0.2695666306	0.2695666306	1.3085852738
5:	0.3072670803	0.2263771909	0.4555088327	0.5741192369	0.5985608690	0.3072670803	0.3072670803	1.6739896317
6:	0.319592227	0.2370589847	0.4982360102	0.6242415033	0.6380013409	0.3195922270	0.3195922270	2.0264888464
...:	.....							
...:	.....							
66:	0.3705996545	0.2550032147	0.5700129302	0.739514219	0.7785870803	0.3705996545	0.3705996545	4.5449079061
67:	0.3705996552	0.2550032147	0.5700129302	0.739514219	0.7785870803	0.3705996552	0.3705996552	4.5449342588
68:	0.3705996552	0.2550032147	0.5700129302	0.739514219	0.7785870803	0.3705996552	0.3705996552	4.5449566592
69:	0.3705996552	0.2550032147	0.5700129302	0.739514219	0.7785870803	0.3705996552	0.3705996552	4.5449756995
70:	0.3705996552	0.2550032147	0.5700129302	0.739514219	0.7785870803	0.3705996552	0.3705996552	4.5449918831
71:	0.3705996552	0.2550032147	0.5700129302	0.739514219	0.7785870803	0.3705996552	0.3705996552	4.5450056392
72:	0.3705996552	0.2550032147	0.5700129302	0.739514219	0.7785870803	0.3705996552	0.3705996552	4.5450173323
73:	0.3705996552	0.2550032147	0.5700129302	0.739514219	0.7785870803	0.3705996552	0.3705996552	4.5450272715
74:	0.3705996552	0.2550032147	0.5700129302	0.739514219	0.7785870803	0.3705996552	0.3705996552	4.5450357190
.	.....							.....
.	.....							.....
.	.....							.....
130:	0.3705996552	0.2550032147	0.5700129302	0.739514219	0.7785870803	0.3705996552	0.3705996552	4.5450835874
131:	0.3705996552	0.2550032147	0.5700129302	0.739514219	0.7785870803	0.3705996552	0.3705996552	4.5450835881
132:	0.3705996552	0.2550032147	0.5700129302	0.739514219	0.7785870803	0.3705996552	0.3705996552	4.5450835888
133:	0.3705996552	0.2550032147	0.5700129302	0.739514219	0.7785870803	0.3705996552	0.3705996552	4.5450835895
134:	0.3705996552	0.2550032147	0.5700129302	0.739514219	0.7785870803	0.3705996552	0.3705996552	4.5450835902
135:	0.3705996552	0.2550032147	0.5700129302	0.739514219	0.7785870803	0.3705996552	0.3705996552	4.5450835902

Figure 31: Result after Applying the Existing PageRank Algorithm on Web Graph shown in figure 30

Case 2: Run the Web Graph on Proposed Modified Algorithm

Figure 32 shows the result after applying the web graph shown in figure 30 on the modified algorithm. Now the number of iteration is 68. The value in green color shows the number of iteration. From the diagram it can be observed that the algorithm is now converged at the 68th iteration. From the figure it is clear that the page rank value of all the nodes except hyothteical node is same in both the iteration at the 67th and 68th (value in red color). The value of hypothetical node at 67th iteration is 4.544342588 and the value of hypothetical node at 68th iteration is 4.544566592. But the algorithm converge at the 68th iteration when modified algorithm is run on the web graph(shown in figure 30.

1:	0.1499999765	0.1499999765	0.1499999765	0.1499999765	0.1499999765	0.1499999765	0.1499999765	0.1499999765
2:	0.1924999702	0.1924999702	0.3199999542	0.3199999542	0.2774999598	0.1924999702	0.1924999702	0.5324999271
3:	0.2286249667	0.2045416357	0.3681666149	0.4765416045	0.4219999446	0.2286249667	0.2286249667	0.9298748856
4:	0.2695666306	0 .2147770517	0.4091082788	0.5277186830	0.5550603515	0.2695666306	0.2695666306	1.3085852738
5:	0.3072670803	0.2263771909	0.4555088327	0.5741192369	0.5985608690	0.3072670803	0.3072670803	1.6739896317
6:	0.3195922270	0.2370589847	0.4982360102	0.6242415033	0.6380013409	0.3195922270	0.3195922270	2.0264888464
7:	0.3307670278	0.2405511100	0.5122045104	0.6640517303	0.6806052688	0.3307670278	0.3307670278	2.3456690869
8:	0.3428381409	0.2437173035	0.5248692854	0.6790911496	0.7144439629	0.3428381409	0.3428381409	2.6294391863
.	.....							
.	.....							
.	.....							
.	.....							
60:	0.3705996531	0.255003214	0.5700129274	0.7395142148	0 .7785870746	0.3705996531	0.3705996531	4.5446177799
61:	0.3705996531	0.255003214	0.5700129281	0.7395142162	0.7785870760	0.3705996531	0.3705996531	4.544687649
62:	0.3705996538	0.255003214	0.5700129281	0.7395142169	0.7785870774	0.3705996538	0.3705996538	4.5447470377
63:	0.3705996545	0.255003214	0.5700129288	0.7395142169	0.7785870781	0.3705996545	0.3705996545	4.5447975191
64:	0.3705996545	0.2550032147	0.5700129302	0.7395142176	0.7785870781	0.3705996545	0.3705996545	4.5448404291
65:	0.3705996545	0.2550032147	0.5700129302	0.7395142190	0.7785870788	0.3705996545	0.3705996545	4.5448769033
66:	0.3705996545	0.2550032147	0.5700129302	0.7395142190	0.7785870803	0.3705996545	0.3705996545	4.5449079061
67:	0.3705996552	0.2550032147	0.5700129302	0.7395142190	0.7785870803	0.3705996552	0.3705996552	4.5449342588
68:	0.3705996552	0.2550032147	0.5700129302	0.7395142190	0.7785870803	0.3705996552	0.3705996552	4.5449566592

Figure 32: Result after Applying the Modified Algorithm on Web Graph shown in figure 30

The number of iteration in each case is shown in table 3. By comparing the number of iterations in both the cases it can be seen that the modified algorithm takes less number of iteration than it

takes previously to converge the algorithm. In case 1 it takes 135 iteration but in case 2 it only takes 68 iteration.

Table 3: Number of iterations in both the Cases in second Experiment

Case	Number of iteration
Case 1	135
Case 2	68

From both the experiment it is clear that in each experiment the number of iteration in case 2 takes less number of iteration than case 1. So the computational problem which is increased due to hypothetical node is decreased with this modified algorithm. So all the issue related to dangling pages, philosophical issue is solved by including the dangling pages in page rank computation and computational problem is solved with the proposed modified algorithm using hypothetical node.

Chapter 6

Conclusion and Future Scope

6.1 Conclusion

The growth of the web is increasing speedily and accessibility has created the need for returning the best result like query relevant pages on the top of list. For obtaining the best results PageRank algorithm is used by Google. In this thesis first the PageRank algorithm and dangling pages are discussed. This thesis also covers all the issues associated with dangling pages philosophical issue and computational issue. From the existing PageRank algorithm the dangling pages are removed from the computation. Many previous approaches to handle theses dangling pages are also discussed. After applying the hypothetical node approach the dangling pages are included in the PageRank calculation so the philosophical issue is solved but the computational problem is increased too much by this approach. In this thesis a modified algorithm to handle dangling pages using the hypothetical node in page rank computation is proposed. Which solve all the issue related to dangling without increasing the computational problem. Experimental results are also shown for better understanding. The implementation of this algorithm produce accurate page rank value of dangling pages and non-dangling pages and produce more accurate ranking of pages. The algorithm is now converged in few iteration as compare to the existing PageRank algorithm. The modified algorithm handles the dangling pages with the best of our knowledge. At last the modified algorithm and existing algorithm is run on two web graph and the value of number of iteration in each case is compared. The modified algorithm takes less number of iteration on the web graph as compare to existing PageRank algorithm.

6.2 Future Scope

The future work may involve further modification in the implementation of this algorithm to optimize its performance and work can be extended to reduce the complexity of this algorithm. The future work also involves how the page rank value of each node can be computed from matrix computation by dividing the computation into different categories like computing the dangling nodes and non-dangling nodes separately.

References

[1] Aly M.M.R. and Khedr A.E., "An integration Framework for Search Engine Architecture to Improve Information Retrieval Quality", in *Proceedings of 2nd International Conference on Computer Technology and Development (ICCTD)*, pp. 506 – 510, 2010.

[2] Baeza Yates R., Boldi P., and Castillo C. "Generalizing PageRank: Damping Functions For Link-Based Ranking Algorithms", in *Proceedings of 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 308-315, August 2006.

[3] Best Search Engine. Available at: http://www.addaycampus.com/2011/12/top-10-best-search-engine.html.

[4] Bianchini M., Gori M. and Scarselli F., "Inside PageRank", *ACM Transactions on Internet Technology*, vol. 5, no.1, pp. 92-128, 2005.

[5] Boldi P., Santini M., and Vigna S., " PageRank as a function of the damping factor" , in *Proceedings of the 14th international conference on World Wide Web*, pp. 557-566, May 2005.

[6] Borodin A., Roberts G. O., Rosenthal J. S. and Tsaparas P., "Finding authorities and hubs from link structures on the World Wide Web", in *Proceedings of the 10th International World Wide Web Conference*, pp. 415-429, 2001.

[7] Borodin A., Roberts G. O., Rosenthal J. S. and Tsaparas P., "Link Analysis Ranking Algorithms, Theory, and Experiments", *Journal of ACM Transactions on Internet Technology*, vol. 5, no. 1, pp. 231-297, February 2005.

[8] Brin S. and Page L., “The Anatomy of a Large-Scale Hypertextual Web Search Engine”, in *Proceedings of the Seventh International World Wide Web Conference*, pp. 107-117, 1998.

[9] Brin S., Motwani R., Page L. and Winograd T., "The Pagerank Citation Ranking: Bringing order to the Web”, *Stanford Digital Library Technologies Project*, November 1999.

[10] Brin, S., Motwani, R., Page, L. and Winograd T., “What can you do with a Web in your pocket?”, *Bulletin of the Technical Committee on Data Engineering*, vol. 21, no. 2, pp. 37-47, 1998.

[11] Chakrabarti S., Dom B., Gibson D., Kleinberg J.M., Raghavan P. and Rajagopalan S., "Automatic Resource Compilation by Analyzing Hyperlink Structure and Association Test", in *Proceedings of The Seventh International World Wide Web Conference*, pp. 65-74, 1998.

[12] Degen D., “Google Page Rank Algorithms for Data Base Systems”, May, 2007.

[13] Dunham M.H., “Data Mining Introductory and Advanced Topics”, Pearson Education, 2003.

[14] Eiron N., McCurley K. and Tomlin. J., "Ranking the Web Frontier", in *Proceedings of the 13th International Conference on World Wide Web*, pp. 309-318, 2004.

[15] Etzioni O., “The world wield web: Quagmire or Gold Mining”, *Communications of the ACM*, vol.39, no.11, pp. 65-68, November 1996.

[16] Fagin R., Kumar R., and Sivakumar D., “Comparing top k lists”, in *Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms*, pp. 28-36, 2003.

[17] Gibson D., Kleinberg J., and Raghavan P., “Inferring Web Communities from Link Topology” in *Proceedings of 9th ACM Conference on Hypertext and Hypermedia*, pp. 225-234, 1998.

[18] Haveliwala T., “Efficient computation of PageRank”, Technical Report 1999-31, Stanford InfoLab, 1999.

[19] HITS Algorithm, Available at http://en.wikipedia.org/wiki/HITS_algorithm.

[20] Ipsen C.F. and Selee T.M., “PageRank Computation with Special attention to Dangling nodes”, *SIAM Journal on Matrix Analysis and Applications*, vol. 29, no. 4, pp. 1281-1296, 2007.

[21] Jaegel W. and Smyth G., “The Importance of Search Engines”, Available at http://www.inetasia.com/NewsandEvents/importance-of-search-engine.pdf.

[22] **Jazi M. D. and Ajoudanian S., "Deep Web Content Mining", World Academy of Science, Engineering and Technology, vol. 49, pp. 501 -505, 2009.**

[23] Kamvar S., Haveliwala T., Manning C., and Golub G., “Exploiting the Block Structure of the Web for Computing PageRank”, Stanford University, Technical Report, 2003.

[24] Kleinberg J., “Authoritative Sources in a Hyperlinked Environment”, *Journal of the ACM*, vol. 46, no. 5, pp. 604-632, September 1999.

[25] Kosala R. and Blockeel H., "Web Mining Research: A Survey", *ACM SIGKDD (Special Interest Group (SIG) on Knowledge Discovery and Data Mining)*, vol. 2, no.1, pp.1-15, June 2000.

[26] Langville A. N. and Meyer C.D., “A Reordering for the PageRank problem”, *SIAM Journal on Scientific Computing*, vol. 27, no. 6, pp. 2112-2120, 2006.

[27] Langville A.N. and Meyer C. D., "A Survey of Eigenvector Methods of Web Information Retrieval", *Journal* of *SIAM*, vol. 47, no.1, pp. 135-161, 2005.

[28] Langville A.N. and Meyer C.D., “Deeper Inside PageRank”, *Internet Mathematics*, vol. 1,

no. 3, pp. 335-380, 2005.

[29] Langville A.N. and Meyer C.D., “Google’s PageRank and Beyond: The Science of Search Engine Rankings”, Princeton University Press, Princeton, NJ, USA, 2006.

[30] Lee C. P., Golub G. H., and Zenios S. A., “A Fast Two-Stage Algorithm For Computing PageRank and its Extensions”, Technical report, Stanford University, 2003.

[31] Lee C.P., Golub G.H. and Zenios S.A., “Partial State Space Aggregation Based on Lump ability and its application to PageRank” Technical report, Stanford University, 2003.

[32] Lee L. “Page Rank Algorithm and Development”, February 2010.

[33] Lempel R. and Moran S., “The stochastic approach for link-structure analysis (SALSA) and the TKC effect”, *International Journal of Computer and Telecommunications Networking*, vol. 33, no. 1-6, pp. 387-401, June 2000.

[34] Liu B., Ma Y., and Philip S. Y., “Discovering business intelligence information by comparing company Web sites”, in: Zhong N., Liu J. and Yao (eds.) Y. Y., “Web Intelligence”, Springer Verlag, pp. 105-127, 2003.

[35] Madria S.K., Bhowmich S.S, Ng W.K., and Lim E.P., “Research issues in Web Data Mining”, in *Proceedings of the First International Conference on Data Warehousing and Knowledge Discovery*, pp. 303-312, 1999.

[36] Manning C. D., Schütze H., Raghavan P., "Introduction to Information Retrieval", Cambridge University Press, New York, USA, 2008.

[37] Marchiori M., “The quest for correct information on Web: Hyper Search Engines”, in *Proceedings of the 6th International World Wide Web Conference*, April 1997.

[38] Norris J., “Markov Chains”, Cambridge University Press, 1996.

[39] Paul G., “the 10 Best Search Engines of 2012”, Available at http:// topbestlisted.blogspot.in/ 2011/08/ worlds-top-10-best-popular-search.html.

[40] Ramakrishna M.T., Gowdar L.K., Havanur M.S. and Swamy B.P.M., “Web Mining Accomplishments & Future Directions”, in *Proceedings of International Conference on Data Storage and Data Engineering (DSDE)*, pp. 187-191, 2010.

[41] Sargolzaei P., Soleymani F. “PageRank Problem, Survey And Future Research Directions”, *International Mathematical Forum*, vol. 5, no. 19, pp. 937 – 956, 2010.

[42] Search Engine, Available at http://www.webopedia.com/TERM/S/search_engine.html.

[43] Singh A.K., Kumar P.R. and Leng A. G. K. “Efficient Algorithm for Handling Dangling Pages Using Hypothetical Node”, in *Proceedings of 6th International Conference on Digital Content, Multimedia Technology and its Applications (IDC)*, pp. 44-49, 2010.

[44] Singh B. and Singh H.K., “WEB DATA MINING RESEARCH: A SURVEY” in *Proceedings of International conference on Computational Intelligence and Computing Research (ICCIC)*, pp. 1-10, December 2010.

[45] Srivastava J., "Web Mining: Accomplishments & Future Directions", *National Science Foundation Workshop on Next Generation Data Mining NGDM02*, pp. 1-148, 2002.

[46] The size of the World Wide Web: Estimated size of Google's index, Available at http://www.cirrusabs.com/blog/all-about-google-caffeine.

[47] Thelwall M., “Mining the World Wide Web: an Information Search Approach”, *Journal of Documentation*, vol. 58, no.2, pp. 232-234, 2002.

[48] Wang, Tao X. T., Sun J. T., Shakery A. and Zhai C "DirichletRank: Solving the Zero-One-Gap Problem of PageRank”, *ACM Transactions on Information Systems*, vol. 26, no.2, pp. 1-29, 2008.

[49] Webtrends, Available at http://www.webtrends.com.

[50] Zhang H., Goel A., Govindan R., Mason K., and Roy B. V., "Making Eigenvector-Based Reputation Systems Robust To Collusion", *Workshop on Algorithms and Models for the Web Graph (WAW)*, May 2004.

List of Publication

1. Srivastava Shipra and Aggrwal Rinkle Rani, "A Modified Algorithm to Handle Dangling Pages Using Hypothetical Node", International Journal of Computer Application. **[Communicated].**

2. Srivastava Shipra, Aggrwal Rinkle Rani and Cheema Karamjit, "A Comprehensive Study of Approaches to Handle Dangling Nodes in Web Page Rank Computation", International Conference On Advanced Computing Technologies. **[Accepted].**