Enhanced Ranking and Storage of search results using Markov Chain

Swati Jain, Ankit Dubey, Mukesh Rawat

Abstract*: Much of the time, we need to retrieve the most excellent response to the data available on various links of archives when given a search query. A practical application of Information Retrieval techniques is a Search Engine which helps in finding the relevant content for the input query. In this paper, we propose an ideal technique for analysis of the hyperlinks by taking the Web as a graph structure and we center around the utilization of hyperlinks for ranking web indexed lists and also provide a technique for storage of search results. This paper also includes various Page Ranking algorithms along with their drawbacks.

Index Terms: Link Analysis, Markov Chain, Page Ranking, Search Engine, Webgraph.

I. INTRODUCTION

A Search Engine is an online apparatus which alludes to a gigantic database of web assets that empowers clients to find data on the World Wide Web. To search for information, clients can scan data by sending catchphrases or keywords as input query, and after that search engine gives response within few sub-second times to extensive amount of user queries presented each day all around the globe [2]. The results of searching provided by search engine consist of the sequence of responses which is termed as Search Engine Result Pages (SERPs). The results might be a mix of site pages, images and various types of records.

A. Components of Search Engine

Mainly, a search engine has three fundamental components which are explained as:

1. Web Crawler

The primary responsibility of a crawler is to identify and acquire documents to be processed by search engine. A web crawler is intended to pursue the connections on website pages to find new pages and download those pages. It is basically a software segment that visits web to accumulate information. Another synonym for it is **spider** or **bots**.

2. Database

A database stores whole of the web information. It is a repository which consists of huge web resources such as web documents, audios, pictures, videos and various other files.

3. Search Interfaces

Search Interface acts as a bridge between the end-user and database repository. It assists end-user to look within the database to find the relevant information.

B. Architecture of Search Engine

The search engine searches for catchphrase by looking index record in database which is predefined rather than reaching to the web directly to find input catchphrase. Afterwards, it makes use of software for searching information stored in database. Such component is referred as web crawler. When web crawler indentifies pages, the most relevant and informative web pages are shown as results. The content of such pages commonly has page's title, text matter size, initial sentences and so on. Fig 1 depicts architecture of a search engine. It consists of two processors: Front-end Processor and Back-end Processor.

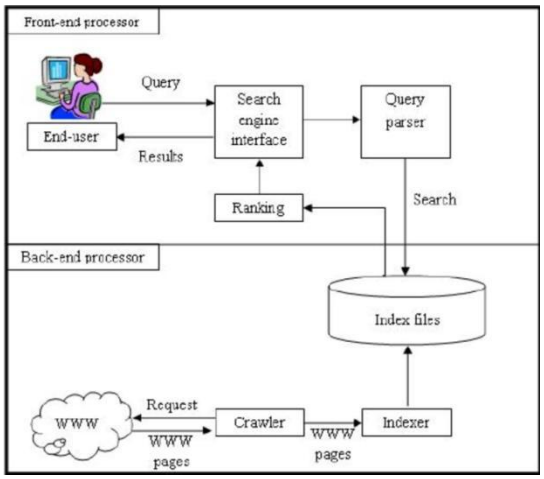


Fig 1: Architecture of Search Engine

1. Front-end Processor: In this, firstly the end-user enters a search query in the search engine interface, which is commonly a web page having a text-box. The query-parser then parses the request so that it can be understand by the engine. After this, the search engine runs the search in the index files and the ranking module ranks the web pages which are displayed to the end-user.

Revised Manuscript Received on December 22, 2018.

[1]**Swati Jain,** [2]**Ankit Dubey,** M.Tech Scholar, Department of Computer Science and Engineering, Meerut Institute of Engineering and Technology, Meerut, India.

Mukesh Rawat, Professor, Department of Computer Science and Engineering, Meerut Institute of Engineering and Technology, Meerut, India.

2. Back-end Processor: The crawler retrieves the web pages from the world wide web and then the indexer parses the web pages which are stored in the index files.

C. Ranking Module

The input to the algorithms of search engine has the main parts of any website page, which involves title, text part and density of keyword, and then finally gives a ranking to every web page which tells the positioning to put the pages as a result [7]. Ranking in web search tool alludes to a site's situation in the internet searcher results page. There are different ranking variables that impact whether a site seems higher on the SERP dependent on the content significance to the pursuit term, or the nature of connections indicating the page. Each web index gives distinctive weights to these ranking variables that is reason when we enter a similar pusuit term in various web crawlers we will normally get diverse outcomes. The ranking is given to every page of website and not to the website as a whole.

The components which influence ranking in a search engine are:

Scoring: This component is also known as query processor, compute scores of web pages by making use of ranking procedure, which depends on retrieval model. Every ranking procedure is inherently depends on such model. Most common form to find the score of every web documents is given as:

$$\sum_{j} t_j \,.\, w_j$$

where t_j is the jth weight of query term, and w_j is the jth weight of document term and summation is done for each term present in the dictionary of collection of repository.

Performance Optimization: This component includes the architecture of ranking procedures and also related indexes in order to minimize result processing time and maximize throughput of query. The scores of any document must be computed and contrasted so quickly so as to decide the ordering of web documents which are provided as the final output to the end user. This is the primary responsibility of performance optimization component.

Distribution: When given some type of index dispersion, distribution of ranking can also be done. An inquiry dealer chooses and identifies the allocation strategy of queries to multiple processors present in a network and holds the responsibility to gather list which contains ranking. The task of dealer relies on the type of index dispersion. The distribution can also be done by using caching where indexes as well as ranking lists of earlier query is put in local space.

II. RELATED WORK

We survey some related work for page ranking using Markov Model as well as Hidden Markov Model(HMM). Wenxing Ye [9] analysed the PageRank Algorithm using the Markov Chain collection as a main device to evaluate various parameters of the page ranking. He showed that extra valuable data could be obtained by applying PageRank to the aggregated Markov Chain and proposed a better way to rank pages using dynamics of Markov Chain.

Prerna Rai [10] proposed that for handling huge amount of documents which is collected in web, Mapper Reducer functions are used and for the same HMM could be integrated along with the Mapper Reducer to accomplish parallelization.

Interpretation of PageRank algorithm with Markov chain could be discovered by making a search on Google using the two main keywords "Markov Chain" and "PageRank". Markov Chain's Stochastic stability is described in [11][12].

III. RANKING ALGORITHM FOR LINK ANALYSIS

A. HITS Ranking Algorithm

Jon Kleinberg proposed an algorithm called Hyperlink-Induced Topic Search abbreviated as HITS and also termed as authorities and hubs. This algorithm is used for analysis of links which ranks the pages. The thought following Authorities and Hubs originated from a specific knowledge into the making of site pages; that is, some site pages, called as hubs, filled in as vast registries which were not actually definitive in data they contain, yet utilized as compositions of a broad list of data which took clients direct to new authentic pages. Or we can say that, good hub shows a webpage which indicated numerous new pages, and good authority shows a webpage which was connected by a wide various other hubs.

In this scheme, when a query is given, two scores are allotted to each web page. Among both scores, first refers to hub score and second refers to authority score. For every input query, two lists of ranking are computed instead of single. The first rank list is actuated through their hub scores whereas second by authority scores.

The HITS algorithm starts by fetching best pages according to the given input query. The web pages so obtained are called root set & it may be retrieved by considering only good pages provided through text-based searching algorithm. Then the root set is enhanced to form the base set for all web pages that links from them and some webpages which links to it. A focused subgraph is created by using pages from base set and all hyperlinks in web pages. The computation of HITS algorithm is performed on the focused subgraph. As indicated by Kleinberg, main purpose to generate base set guarantees that almost all capable authorities are considered.

The hub and authority scores values are defined in a shared iteration. The computation of authority score is done by total of all values of scaled hub which links to particular page. And, the value of hub score is the total of all values of scaled authority of pages which links to. Some applications take its importance to the connected web pages.

This algorithm basically consists of two steps which are performed iteratively, which includes:

- **Authority Update**: For each node, authority score is updated which is equal to the sum of each node's hub score which points to it. That is, a high authority score is given to the node by being connected from pages that are perceived as Hubs for relevant information.
- **Hub Update:** For each node, hub score is updated which is equal to the sum of each node's authority scores which points to it. That is, a high hub score is given to the node by connecting to nodes that are taken to be authorities on that node.

The calculation of Authority and Hub score of every node are explained in below steps:

- First, the process is started from each node which has a authority and hub score set to 1.
- Now apply rule of authority update
- Then perform update of hub
- Then values are normalized as dividing every score of Hub to square base of total of squares of every Hub scores, & dividing every score of Authority to square base of entirety of squares of each Authority scores.
- As required, iteration from second step is done.

Merits of HITS Algorithm

- Allow query by topics, which might have capacity to give more appropriate hub and authority pages.
- Allows to build the adjacency matrices from the neighbourhoods graph & iterations in power do not have any computational burdens.

Demerits of HITS Algorithm

- The neighbourhood graphs must need to be constructed in "On-Fly" manner i.e. ranking of hub and authority depends on query. Small changes in web lead to substantial change in hub and authority scores.
- It experiences topic-drift i.e. neighbour graphs might consists nodes having great authority value for an unrelated topic to input query.
- Advertisements can't be detected.
- HITS can easily be spammed by adding outbound links in owner's page.
- Evaluating query is slower. It is because "On-Fly" neighbourhood graphs creation.

B. The PageRank Algorithm

PageRank algorithm is used for analyzing links developed by Larry Page and Sergery Brin. Google searching make use of PageRank algorithm to rank sites as results of search engine query. It is a method of estimating the significance of web pages. A numerical weighting is assigned to every component of all hyperlinked documents, for example, World Wide Web [3][5], with the motivation behind "estimating" its relative significance in the document set.

A PageRank [1] is declared from a scientific calculation dependent on a webgraph, where nodes are made by pages in WWW and edges are links. The significance of any page is indicated by its rank value. A web page's PageRank is characterized repeatedly and relies upon PageRank metric as well as number of every page that connects it ("incoming hyperlinks"). A page that is connected by numerous web pages having high PageRank gets higher position in the result.

A numeric value is taken to express the probability which lies between 0 and 1. A probability of 0.5 is generally denoted as "50% chance" to happen something. Therefore, value 0.5 of PageRank shows having 50% chance of visiting surfer clicks on any arbitrary link which directs towards document link having 0.5 value of PageRank.

- **The Simplified Algorithm**

Suppose a tiny web world having four site web pages say P, Q, R and S. Self loop connections, or various outgoing link from single web page to other single web page is disregarded. The same value is initialized for every page as the PageRank. Original PageRank form takes total of PageRank values for each page was total web pages present web that instant, thus every page in given example takes 1 as its initial value. But, PageRank new versions, considers as the probability lies in range 0 & 1. Therefore, in example the initial value is taken to be 0.25.

The values of PageRank is exchanged from one page to destination of it's outgoing connections upon following iteration is separated fairly to each outgoing connection[8]. Suppose the system has links from web pages Q, R, and S to P, every link will exchange PageRank value of 0.25 to P in next iteration, 0.75 a total value to P.

$$Prob(P) = Prob(Q) + Prob(R) + Prob(S)$$

Now consider that Q page has link to web pages R and P, page R has link to P, and S links to P, Q, R. Hence, during first iteration, web page Q will exchange its half value i.e. 0.125, to P and remaining, (0.125), to R. Web page R transfer its whole value (0.25), to P. Since S contains 3 outgoing connections, it transfers its one-third value, i.e 0.083 approx., to P. After completion of current iteration, web page P consists of PageRank value of 0.458 approximately.

$$Prob(P) = Prob(Q)/2 + Prob(R)/1 + Prob(S)/3$$

Or we can say that, PageRank value deliberated through outgoing link is same as PageRank score of own document divided to number of outgoing connections C( ).

$$Prob(P) = Prob(Q)/C(Q) + Prob(R)/C(R) + Prob(S)/C(S)$$

Generally, the value of PageRank of a page 'a' is given as:

$$Prob(a) = \sum_{b \in Za} Prob(b)/C(b)$$

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

which means that value of PageRank of any page 'a' depends on every web page 'b' PageRank value that is a part of set Za (it is the set of web pages which links to web page 'a') and divided by C(b) number of links from web page 'b'.

Advantages of Page Rank

- Provide robustness against Spamming because adding inlinks towards a web page from relevant pages is not an easy task.
- Computation of PageRank is completely query independent.

Disadvantages of Page Rank

- Its main disadvantage is that it tends to favour older web pages, this is so due to a new web page, even the most appropriate one do not contain various links until it belongs to an edge of existing one.
- It is easy to increase the value of PageRank due to concept "link-farms" which a collection of websites that link to all of the website in group. But, while indexing process can easily identify these problems.

IV. PAGERANK USING MARKOV CHAIN

Consider an arbitrary surfer who starts from a page and then executes an arbitrary tour on web. If the present position of the arbitrary surfer contains no outbound links, a teleport operation is used. During teleport operation, arbitrary surfer moves from one node to another node present in web graph just by writing a web address into browser's URL. While assigning value of PageRank score to every vertex of webgraph, in two forms the teleport operation is used: (1) When a node has no outgoing links, teleport operation is executed by surfer. (2) For any node having outbound links, the teleport operation is invoked by surfer having a probability of $0 < \alpha < 1$ and a standard arbitrary tour having probability of $1 - \alpha$, here 'α' is a constant parameter taken in prior.

A. General Architecture

The Markov chain is defined as stochastic discrete time process which is process happens in progression of time ventures in every one of which an irregular decision is done. Generally, Markov chain comprises of 'N' different states.

The characterization of Markov Chain is done by a transition probability matrix (N x N) 'T' and every entry lies in range of [0, 1] and each row entry of 'T' sum to 1. At any instant among N states Markov chain is one of it. Then, entry Tij indicates probability to be in state j at next timestamp, but the current state should be i. Tij entry in matrix refers to transition probability and based on the present state 'i' only; this property is termed as Markov property. Thus, by the Markov property,

$$\forall \text{ i, j, } T_{ij} \in [0, 1]$$

and

$$\forall \text{ i, } \sum_{j=1}^{N} Tij = 1 \qquad (1)$$

The matrix that has non-negative entries and fulfills Equation (1) is termed as "stochastic matrix". The most imporatnt property of such stochastic matrix is it contains principal *left eigenvector* equivalent to its eigenvalue,of 1 which is largest.

A Markov chain"s probability distribution over its states may be viewed as a probability vector: a vector all of whose entries are in the interval [0, 1], and the entries add up to 1. An N-dimensional probability vector each of whose components corresponds to one of the N states of a Markov chain can be viewed as a probability distribution over its states.

B. Implementation Steps

1. Finding Transition Probability Matrix

For any website, create a web graph of the links in the website (the graph is directed graph) [4]. Let there are N nodes in the graph. The probability of visiting a web page by a random surfer is given by *Transition Probability Matrix* (T).

Step1: First create the adjacency matrix 'A' of the web-graph as follow:
If there is a hyperlink from page *i* to page *j* then $A_{ij}=1$, otherwise $A_{ij}=0$.

Step2: The Transition Probability matrix (T) is derived as:
(i) If a row of A has no 1entries, then change every entry in matrix to 1/N. For remaining rows do the following.
(ii) For each row having 1 entries, divide every 1 to total number of 1's in its row. Thus, if there is a row with three 1's, replace every 1 to 1/3.
(iii) Changed matrix is multiplied by $(1-\alpha)$
(iv) Finally, add (α/N) to resulting matrix entries, & T is obtained.

As size of website (or the number of hyperlinks) increases, the connected nodes within a specific webgraph also increases which leads to the creation of a very large size matrix, which is difficult to maintain in memory. Therefore, alternative strategy should be adopted to accumulate such an increased matrix of webgraph.

To overcome this problem, a matrix is maintained in a file system (excel form) which can store such a huge information as compared to the matrix stored in the main memory.

Consider the sample webgraph for a website and the corresponding Transition Probability Matrix (created and maintained in an excel file) assuming $\alpha = 0.5$:

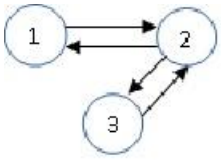
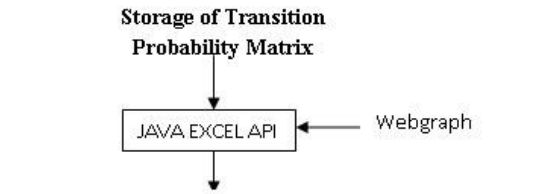


Fig 2: Webgraph



Nodes	1	2	3
1	1/6	2/3	1/6
2	5/12	1/6	5/12
3	1/6	2/3	1/6

Fig 3: Storage of Matrix in Excel

The Transition Probability Matrix created is stored in the excel sheet by using the java excel API, which provides a large storage to the indexes.

2. API used to access excel

jxl-2.6.12 api is used to write matrix information to the excel, which provides a large storage ,easily viewable and accessible of information because the data is maintained in the cells of the excel sheet in an organized manner. The code below show how to create a excel sheet from application and create columns and feed data into cells.

```
WritableWorkbook wo = Workbook.createWorkbook(new File("output.xls"));
WritableSheet sh = workbook.createSheet("First Sheett",0);
   JLabel l1 = new JLabel(0,1,"term");
   JLabel label1 = new JLabel(0,2,"Doc id"s");
   sh.addCell(label);
   sht.addCell(l1);
   jxl.write.Number num = new jxl.write.Number(1,2,1);
   sheet.addCell(number);
   ……
   ……
   workbook.write1();
   workbook.close1();
```

C. Computing Page Rank

The transition probability matrix T has left eigen vectors represented as N-vectors denoted by $\vec{\pi}$ given as:

$$\vec{\pi} T = \omega \vec{\pi} \qquad (2)$$

The 'N' values of principal left eigenvector $\vec{\pi}$ are consistent state probabilities of arbitrary tour having teleport operation enabled, therefore, becomes the values of PageRank of the web documents or pages. The left eigenvectors are computed by using "*iteration in power*" method. Let $\vec{y}$ be beginning distribution for all states, then at time '*t*' the distribution is given by $\vec{y}P^t$. As far as '*t*' value increases, the distribution could be expected as $\vec{y}P^{t2}$ which tends to be likely to the distribution of $\vec{y}P^{t+1}$, as the value of 't' increases we will arrive at a steady consistent state.

The iterations in power technique imitate the walk of a surfer as: start from any state and executes arbitrary walk for an expansive number of timesteps 't', monitoring the number of times each state being visited. As time grows large and value of time steps 't' increases, such frequencies "settle down" such that any change in evaluated frequencies lies beneath some predefined set threshold value. We express these classified frequencies as the values of PageRank. The distribution for the steady consistent state is given in the form $\vec{\pi}$ = (x 1 – 2x x).

For above given webgraph, assume the arbitrary surfer begins from state '1', the starting probability vector is taken to be $\vec{y}_0$ = (1 0 0). On the completion of initial step distribution becomes:

$$\vec{y}_0 = (1/6 \;\; 2/3 \;\; 1/6) = \vec{y}_1$$

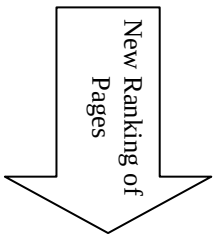
Continuing this way, this distribution gathers to **steady state** given by $\vec{y}$ = (5/18 4/9 5/18) which is the PageRank value of each node of webgraph.

V. REFINE THE SEARCH RESULT USING TERMS OF SEARCH QUERY

We can further enhance the web page links ranking in search result by taking into account terms of search query. To achieve this, we find the frequency of the search term(s) in each link of the search results and adding up the frequency with individual page rank. Then, the search result is refined according to this new value of web link pages.

Suppose,Text(Query) = deadlock and the corresponding search result contains three document links as L1.html, L2.html and L3.html (in order, according to page rank values). Now, the new rank is computed as:

Links	Frequency of term (deadlock)	Page Rank Value	New Rank
L1	22	2/3	22.67
L2	30	1/6	30.16
L3	19	1/6	19.16



L2
L1
L3

VI. CONCLUSION

This paper focuses on the techniques of link analysis for any website and the probability of visiting a web-page by any random surfer. The HITS and PageRank techniques for link analysis are discussed, among these two PageRank is more robust. Also, the web page ranking values of PageRank (along with implied ordering for them) does not dependent on any end user's input search query. Due to this reason, some static quality measures are used by search engines like PageRank as one of the major factors in calculating the score of a web document when a search query is provided. Following values of PageRank for web-pages, the ranking of the pages can be further enhanced by using two-way filtering which includes analyzing the pages that contains maximum search keywords given in the query. Also, to overcome the problem of limited storage space in the server, the adjacency matrix information could be stored at the client side.

REFERENCES

1. N. Duhan, A. K. Sharma and K. K. Bhatia, "Page Ranking Algorithms: A Survey", Proceedings of the IEEE International Conference on Advance Computing, 2009.
2. Dinesh Sharma, A.K. Sharma, Komal Kumar Bhatia, "Search engines: a comparative review",NGCIS-2007.
3. A. K. Sharma, Komal Kumar Bhatia: "Automated Discovery of Task Oriented Search Interfaces through Augmented Hypertext Documents", accepted at First International Conference on Web Engineering & Application (ICWA2006).
4. A. Broder, R. Kumar, F Maghoul, P. Raghavan, S.Rajagopalan, R. Stata, A. Tomkins, J. Wiener, "Graph Structure in the Web", Computer Networks: The International Journal of Computer and telecommunications Networking, Vol. 33, Issue 1-6 2000.
5. S. Chakrabarti, B. Dom, D. Gibson, J. Kleinberg, R. Kumar, P. Raghavan, S. Rajagopalan, A. Tomkins, "Mining the Link Structure of the World Wide Web", IEEE Computer Society Press, Vol 32, Issue 8 pp. 60 - 67, 1999.
6. L. Page, S. Brin, R. Motwani, and T. Wino grad, "The PageRank Citation Ranking: Bringing order to the Web", Technical report, Stanford Digital Libraries SIDL-WP1999-0120, 1999.
7. S. Lawrence and C. L. Giles, "Context and Page Analysis for Improved Web Search", IEEE Internet Computing, 2(4):38-46, 1998.
8. Gyanendra Kumar, Neelam Duhan, A. K. Sharma, "Page Ranking Based on Number of Visits of Links of Web Page", IEEE International Conference on Computer & Communication Technology (ICCCT)-2011.
9. Wenxing Ye, "On PageRank Algorithm and Markov Chain Reduction", Department of Computer Science and Engineering, University of Florida, Gainesville, FL, USA.
10. Prerna Rai and Arvind Lal, "Google PageRank Algorithm: Markov Chain Model and Hidden Markov Model", International Journal of Computer Applications (0975-8887), Vol. 138, Issue 9, 2016.
11. Meyn, Sean and Tweedie, R. L. (eds.), "Markov Chains and Stochastic Stability", Cambridge University Press,New York, NY, USA, 1st edition, 2009.
12. Niranjan Lal, Shamimul Qamar, Savita Shivani, "Search Ranking for Heterogeneous Data over Dataspace", Indian Journal of Science and Technology (Scopus Indexed). Volume 9, Issue 36, pp.1-9, 2016