IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 19, Issue 1, Ver. III (Jan.-Feb. 2017), PP 01-07
www.iosrjournals.org

PageRank Algorithm

Albi Dode[1], Silvester Hasani[2]
[1]*(Student, School of Innovation, Design and Engineering, Mälardalens Högskola, Sweden*)
[2]*(Student, Computer Science Department, University of New York Tirana, Albania)*

Abstract: *The way in which the displaying of the web pages is done within a search is not a mystery. It involves applied math and good computer science knowledge for the right implementation. This relation involves vectors, matrixes and other mathematical notations. The PageRank vector needs to be calculated, that implies calculations for a stationary distribution, stochastic matrix. The matrices hold the link structure and the guidance of the web surfer. As links are added every day, and the number of websites goes beyond billions, the modification of the web link's structure in the web affects the PageRank. In order to make this work, search algorithms need improvements. Problems and misbehaviors may come into place, but this topic pays attention to many researches which do improvements day by day. Even though it is a simple formula, PageRank runs a successful business. PageRank may be considered as the right example where applied math and computer knowledge can be fitted together.*

Keywords: *Algorithms, Distribution, Google, Math, PageRank.*

I. Introduction

PageRank is a topic widely discussed by Search Engine Optimization (SEO) experts. The heart of PageRank is a mathematical formula which seems to be very complicated, but it is actually simple to understand if simplified. It is necessary for the web developers to understand the concept of PageRank to make the site more high ranked. There is a majority of websites available in the market today. If a person sits and calculates the PageRank for different websites, this may be a tedious work. Since automation is making the life of a person easier, automating PageRank will have a better effect on the World Wide Web. PageRank is an algorithm that drives Google. It was invented by Larry Page and Sergey Brin while they were graduate students at Stanford University and later it became a trademark of google in 1998. PageRank does not have an impact only in the programming industry, but also have an effect in the economic sector. It is also considered as a business goal of every firm to be ranked higher in the web page display, that is considered as a SEO strategy. Hyperlink structure along with SEO strategy plays a vital role in the Page ranking.

There is a math behind every algorithm. Matrix and vector can be considered as the main source of many achievements. Since the amount of websites is growing day by day, the web sites have to be ranked using certain algorithms. The specialty of Google's page rank is that it does not allow spams, which are webpages that are coded in such a way to manipulate the ranking results and that go against the guidelines established by Google. It also focuses on the importance of the page when it is pointed by other important nodes. The functioning of PageRank algorithm also realizes the importance of linear equations and graphs. This report will provide a short summary of math behind the page ranking algorithm. And also explains the problems in the current scenario and suggestions for it.

II. Text Ranking Vs. Page Ranking

In the early 90's, the first search engine used *text based ranking systems* that decided which page is relevant based on the text. There are many drawbacks with this approach. Say for example, if a person search with the keyword "Internet", can be problematic. The surfer might get a page with the keyword internet that does not have any information regarding internet in the displayed page. Since the search engine uses the count of occurrence of words in the given query, it doesn't make sense for the most searched page to be displayed first. There might be millions of web pages that have the searched word and when the search engine brings all those pages, it sounds useless for the surfer. Also, the surfer does not have the patience to go through all the pages that contained the searched word to arrive at the page he/she searching for. Usually, the user expects the relevant page to be displayed in the top 20-30 pages provided by search engine.A modern search engine uses the method of providing best results first that are more appropriate than the older text ranking method. One of the most influential algorithm is the Page Rank algorithm which is used by Google search engine. The main idea behind the page rank algorithm is that the importance of a web page is predicted by the pages linking to it. If we create a web page *i* that has a hyperlink connected to page *j* then page *j* is considered as important. On the other hand, if page *j* has a backlink from page *k* (like www.google.com) we can say *k* transfers its control to *j* (i.e., *k* asserts that *j* is important). We can iteratively assign a rank to each page based on the number of pages that points to it.

III. Calculating Pagerank With Matrix Method Using Outlinks

Today there are millions of websites and each holds tons of information [1]. When a web surfer visits a web page, that point acts as the starting point for the listing. By making use of random selection of links, a website can be visited several times by the surfer. Graphs will provide a better visualization to make it more mathematically concrete [2]. If a web page i has outlinks ≥ 1, then for each element from page i to another page j, the element in the row i and column j of matrix H is defined as = 1/ . If there is no link from page i to j then the value of matrix is 0. Thus, each non-zero element in a row sums to 1. A page with no outlinks is named a dangling node. All the elements in the row of a dangling node are set to 0. Consider the following graph with different nodes and links connecting them.

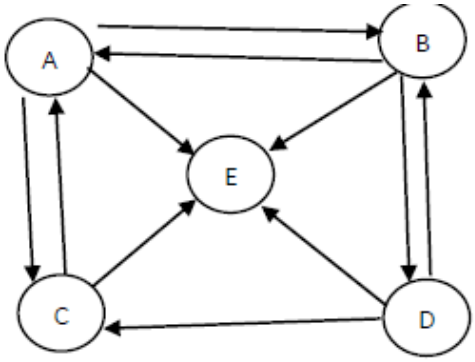

Fig. 1: Directed graph for an illustration to calculate PageRank

The matrix H for the directed graph with the vertex pointing from i to j is given as below.

$$H = \begin{bmatrix} 0 & 1/3 & 1/3 & 0 & 1/3 \\ 1/3 & 0 & 0 & 1/3 & 1/3 \\ 1/2 & 0 & 0 & 0 & 1/2 \\ 0 & 1/3 & 1/3 & 0 & 1/3 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Fig.2: Matrix H

The below table demonstrates the number of inlinks for each and every node.

Table 1: Illustration of the number of links of each node of the diagram

Node	A	B	C	D	E
Number of Inlinks	2	2	2	2	4

In this case, the problem stands on getting spams when a page backlink to other pages. But still, the rank of a page totally relies on backlinks.

This is determined by:

$r(p)=\sum_{p}^{q} \frac{r(q)}{d(q)}$ eq.(1), where the rank value r(p) linking to page p is calculated by rank of the page to the out degree of that page[3].

It is also thought that a link to a page coming from an important page should boost the page`s importance more than a link coming from non-important one.

To illustrate that with our example in the figure 1, we will have these steps:

Step 1. Take a 0.85 * a page's PageRank, and divide it by the number of out links on the page. Step 2. Add that amount onto a new total for each page it's being passed to.

Step 3. Add 0.15 to each of those totals.

As we start at zero, we will have 0.85*0=0. This leads that each page will get 0.15 as 0.15+0=0.15. But still we have the importance based on links. So, now calculations become:

Page A links to pages B, C and E. Page A's PageRank is 0.15 so it will add 0.85* 0.15 = 0.1275 to the new PageRank scores of the pages it links to. There are three of them so they each get 0.0425.

Page B links to page A,C,E. Page B's PageRank is 0.15 so it will add 0.85 * 0.15 =0.1275 to the new PageRank score of the pages it links to. Since it links to page A,C,E, each gets 0.0425. Page C links to Page A,E , each 0.06375. Page D links to Page B,C,E. each 0.0425.Page E links to none.

As a result,
Page A: 0.15 (base) + 0.1275 +0.0425(from Page C,B) = 0.35
Page B: 0.15 (base) + 0.0425+ 0(from Page A,D) = 0.1925
Page C: 0.15 (base) + 0.0425 (from Page A) + 0.0425 (from Page D) = 0.235
Page D: 0.15 (base) + 0.0425 (from Page B) = 0.1925
Page E: 0.15 (base)

IV. Damping Factor

The behavior of PageRank also depends on the function of damping factor, especially what happens when it goes close to 1. Usually, the damping factor is a decay factor. It usually represents when a user gets bored of the current page and stop clicking the current page and move to a random page. This new page may be a new url instead of following from the previous page link. This was initially set to 85% or 0.85. Hence the damping factor is mostly considered to be 0.85. And also there is 15% of chance for the user to visit a random page without following the link from the current page [4].

From that, we can see that this probability of a click-through is a good way to prevent spams and pages without outgoing links and does not allow them to get the PageRank of others. If we use value 1, the forever clicking link process will end up in spam sites and if 0 than we will have random restart and a uniform distribution. In short, whatever the number of inlinks, there will be a probability (1-damping factor) in order for a page to have a minimum PageRank. So, a value in between 0.9 and 0.85 deals with not only making the right calculations but the convergence is not so quick and avoids the growing higher of the PageRank.

V. Calculating Pagerank With Convergence Function Method

PageRank is the mechanism used in Google search engine. PageRank function operates by assigning a value to each and every page that implies the importance of the page. In order to understand the PageRank algorithm let's consider the web as a directed graph G with V as the set of vertices of page n and E as the edges (i,j) if there exists a link from i to j.
PageRank algorithm assigns a rank value r*i* to a page *i* as the function of rank of the page pointing to it.

$$r_i = \sum_{j \to i} \frac{r_j}{d_j}$$

eq.(2),

where di holds the number of the outgoing links of a page *i*. This recursive function gives a rank to the page which is weighted by the out-degree number. In order to numerically calculate the above equation, it is necessary that the graph should possess convergence. Strongly connected graph always has the convergence property, approaching to a limit, and it is achieved by adding some set of outgoing transitions with probability in each page. This makes the above equation to be modified as,

$$r = (1 - c)e + cAr$$

eq.(3),

where e is the vector of all 1's and A is the matrix if there exists a link between i and j and 0 otherwise. Usually, the value of c is considered as 0.85 or 0.9. It is also necessary to remove all rows and columns from the matrix A which has 0 entries. This means that we are removing pages with zero out degree i.e., dangling pages or sink pages. There are cases when the percentage of "follow link" reaches 0, then all pages will nearly have the same PageRank. The intention behind the above PageRank algorithm can be better understood by a random surfer method. When a surfer starts browsing from one web page to the other, the navigation to the other page can be done by following the link provided in the current page uniformly at random. There might be a case when the surfer reaches a page without any outgoing links then he/she may jump randomly to any other page. The PageRank value of a page is determined by the probability that at some time the surfer has reached that page [5].

VI. Linear Algebra Point Of View

To illustrate this algorithm, we begin by identifying the web pages on the internet as a directed graph where the pages are considered as nodes and the hyperlink between the pages are considered as edges. For instance, we have 4 web pages namely 1, 2, 3 and 4. When a website i references to a website j then we add a directed edge between i to j. In our example, we could see that the node 1 has links with all the pages and node 3 has link only with the node 1. A graph for the illustration purpose is depicted as follows,

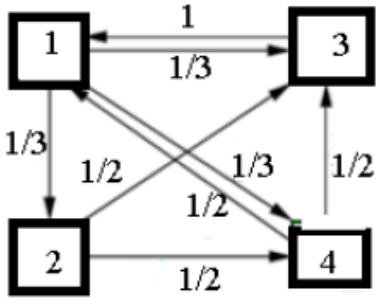

Fig. 3: Directed graph of PageRank calculation using Linear Algebra

The matrix for the above diagram using its outlinks is given as:

$$A = \begin{bmatrix} 0 & 0 & 1 & 1/2 \\ 1/3 & 0 & 0 & 0 \\ 1/3 & 1/2 & 0 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix}$$

Fig. 4: Matrix of the diagram

Let us consider x1, x2, x3 and x4 as the importance of every page. Analyzing the above problem, we can give the importance of each page as follows:

$$x_1 = 1 \cdot x_3 + \frac{1}{2} \cdot x_4$$

$$x_2 = \frac{1}{3} \cdot x_1$$

$$x_3 = \frac{1}{3} \cdot x_1 + \frac{1}{2} \cdot x_2 + \frac{1}{2} \cdot x_4$$

$$x_4 = \frac{1}{3} \cdot x_1 + \frac{1}{2} \cdot x_2$$

Fig. 5: Page Importance

Among the eigenvectors of the matrix A, we pick the one which consists of only non-negative real values. This is the eigenvector corresponding to the eigenvalue λ_1=1, whose entries are:

$$c \cdot \begin{bmatrix} 12 \\ 4 \\ 9 \\ 6 \end{bmatrix}$$

Fig.6: Eigen Vector

Since the PageRank has to calculate the relative importance between nodes and the eigenvalues is just a scalar multiples of each other, we can choose a PageRank vector. This should be the unique values that should make all the entries sums to 1. It is often referred as probabilistic eigenvector corresponding to 1. The eigenvector below is the PageRank Vector.

$$\frac{1}{31} \cdot \begin{bmatrix} 12 \\ 4 \\ 9 \\ 6 \end{bmatrix} \approx \begin{bmatrix} 0.38 \\ 0.12 \\ 0.29 \\ 0.19 \end{bmatrix}$$

Fig. 7: PageRank Vector

VII. Stationary Distribution

The stationary distribution specifies how much time has been spent on an average at a specific node during an infinitely long time random surf. It helps in ordering the web pages. A stationary distribution in a Markov chain is a probability vector *p* i.e., the entries should be greater than or equal to 0 and the sum should equal to 1. This means that the distribution after one step of a Markov chain remains unchanged. Hence we are looking for an eigenvector of A corresponding to the eigenvalue 1.

If a Markov chain is strongly connected (i.e., any state can be reached from any other state) then the stationary distribution *p* exists and its value and it will be unique.

Furthermore, stationary distribution is also considered as proportion of visit the chain pays to every state after a long time (the ergodic theorem).

$$p_i = \lim_{t \to \alpha} \frac{nr.of\ visits\ to\ state\ i\ in\ t\ steps}{t}$$

eq.(4),

where PageRank p_i is termed as the proportion of time our random surfer spends on a webpage *i* when the surfer is allowed to move freely over the webpages.

VIII. Pagerank Convergence As A Function Of The Web Size

There are many factors that define the convergence speed of a PageRank vector. It depends on the random jump probability value, numerical solver and size of the matrix. The following figure shows the result of PageRank convergence with respect to function of the size of a web graph. The figure demonstrates a case when there is a smaller number of pages, as a consequence convergence occurs faster [5].

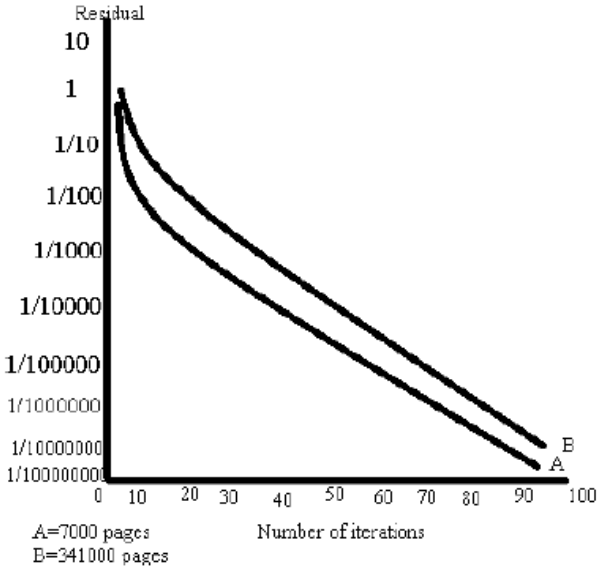

Fig. 8: Convergence as a function of web size

IX. Binary Search For Pagerank

Google goes through the page using inverted indexing technique which uses binary search method. This search is used to retrieve the PageRank of previously calculated pages stored in the repository. It runs on O(log n) time. This shows that it is possible to have a reasonable search time even if we deal with a large amount of data. If the amount of data present in the web is increased then the search time difference will also be increased. There are possibilities for other problems when we deal with a large amount of data. If the system has a large amount of data and those data are indexed then searching is easier. Also inserting a data into sorted index can be done using logarithmic time complexity and so adding new sites to the existing index is also very fast [6].

X. Spam Issue With Pagerank

One of the main issues in PageRank is spamming. Spamdexing is a way used to manipulate search engine indexes. While making use of the PageRank algorithm, some web pages use spams in order to increase the rank of certain pages. When a page receives link from a page which has higher PageRank then the PageRank of current page will also increase. Hence they try to make use of such links to increase the PageRank of their pages. This shows the possibility of sharing page rank in connected nodes.

Inspired by this problem, researchers are focusing in a new area called Personalization. It is an area yet to be explored as the mathematical calculations are hard. A best known technique is that of TrustRank which only used the good pages. They rely on the expansion of a set of good pages. The idea is to gain trust from the good pages and recursively go to the outgoing links. But the problem that arises here is that intruders can leave bad links somewhere in the page. The solution for this is found by inserting a threshold value, if a page is below threshold then it is considered as spam [4].

XI. Storage Issue With Pagerank

Regardless of being a good page or a spam page, the number of web sites keeps growing. For this reason, storing memory is a real issue in PageRank. As a matrix can exceed the capacity of main memory, compressing data may be the solution. In such cases, a modified version of PageRank is used. Or if not, they try to implement I/O efficient computations. Generally, the time complexity of an algorithm is calculated by the number of data processing steps as addition, multiply, etc. But when the data to be worked with are in huge amount and they are bigger even than the main memory, the computational problem becomes more complex. In this case, it will be the number of how many times the disk has been accessed, which will hold importance rather than the time of running. We know that cached data is much faster to be accessed than that in main memory, so the algorithms should be cache friendly somehow. Also, there is another alternative of data compression in order to fit the main memory, but that requires also a modification in the PageRank algorithm. As the PageRank vector has to be consulted in order for the query to be processed, speeding the process requires the help of cache memory. A best-known technique is that of gap method. The idea here, it is that a page has inlinks from pages labeled next to it. For example: if a page is labeled 50, most probably it will have inlinks from pages labeled 49,51. This is the locality principle.

Another idea is that of comparing the similar adjacency list of two pages. Adjacency list of x, contains a 1 in the i-th position if the corresponding adjacency list entry i is shared by x and y. The second vector in the reference encoding is a list of all entries in the adjacency list of y that are not found in the adjacency list of its reference x. But this is not so used as in this case we have the problem of finding what page serves as a reference page to another page. Since the PageRank vector itself is large and completely dense, containing over 8 billion pages, and must be consulted in order to process each user query, has also suggested a technique to compress the PageRank vector. This encoding of the PageRank vector hopes to keep the ranking information cached in main memory, thus speeding query processing [7].

XII. Improvements Of Pagerank

Scientific proposals were made in order to reduce the amount of work in each iteration. Adaptive PageRank was the name given by Kamvar et al. [8] to an improved PageRank. This was due to the fact that convergence comes faster for certain pages. What this method does is the isolation of them in the later computations. It was proven to give a boost of 17%. This is done if redundant calculations such as already converged pages are not considered. Christian Lee is the name of the scientist who divides nodes in dangling and non-dangling and uses aggregation. What this division does is reducing to a $k \times k$ problem, where k is the number of non-dangling nodes on the Web [6].

Researchers have continued the perfection of PageRank by making less iterations and calling it BlockRank. In this version, host categorization of pages is the heart. The link structure is used and still ranking is done locally. After that, the local vectors for the hosts are compared from their importance. In this case, we consider a stationary vector to know how much time the random surfer spends on that host. It was seen that the speedup that was gained was of a factor of 2 [6].Continuing with time as an improvement factor, as days may be required for the computation due to the many web sites, another proposal comes from Gleich. In this new algorithm inner/outer iterations are performed as if the damping factor is smaller, easier it is to solve problems. He applies an iterative scheme in which each iteration requires solving another linear system which is similar in its algebraic structure to the original, but with a lower damping factor [9]. Cevahir goes with parallel computations for reducing the pre-processing of sparse matrix partitions. He goes in two ways: Information available in links and partitioning on distributed data [9].

XIII. Conclusion

So far, we have seen how Google calculates PageRank and how its matrix is a combination of stochastic matrices of link structure and behavior of web surfer. A strong property in the success of this algorithm is that it uses information not on the page content itself but on the linkage connection between pages. As the inspiration was from the ranking of scientific articles, PageRank itself can be thought as a link review. The formula is iteratively calculated and as the web pages keep growing in number and not in a connected manner, a damping constant is used to ease the duty. The mathematic that goes on with eigenvectors and eigenvalues makes it such that it converges. There are many factors that make it work. The best value 0.85, as

analyzed in our article, also this convergence speed gives the best ranking. Another good thing besides removing spamming is that PageRank is query independent. It is an open topic for further improves as storage algorithms and speed improvements are always welcomed, as it is an area for both computer scientists and numerical analysts. Further improvements for the future can be of a PageRank algorithm that can understand the semantic of the sites as more and more complex questions can be answered in a matter of second.

References

[1]. Internet Stats, 2016). Retrieved 2016, from https://internetlivestats.com

[2]. Damping Factor, 2016). Retrieved 2016, from Forum DigitalPoint: https://forums.digitalpoint.com/threads/what-is-the-damping-factor-in-page-rank-and-whats-the-concept-behind-using-it.1885967/

[3]. BIANCHINI, M., GORI, M., & SCARSELLI, F. (2005). Inside PageRank. *ACM Transactions on Internet Technology*, 92-128.

[4]. Srivastava, S., & Aggrawal, R. R. (2012). A Modified Algorithm to Handle Dangling Pages using. *International Journal of Computer Applications*.

[5]. Chen, Y.-Y., Gan, Q., & Suel, T. (2002). *I/O Efficient Techniques for Computing Pagerank.*

[6]. Langville, A. N., & Meyer, C. D. (2004). *Deeper Inside PageRank.* Retrieved from http://meyer. math.ncsu. edu/meyer/ps_ files/ deeperinsidepr.pdf

[7]. Tibshirani, R. (2013). Retrieved from PageRank: http://www.stat.cmu.edu/~ryantibs/datamining/lectures/03-pr.pdf

[8]. Hareshkumar, N., & Garg, D. D. (2011). Random Web Surfer PageRank Algorithm. *International Journal of Computer Applications*.

[9]. Xie, Y., Huang, T.-Z., Wen, C., & Wu, D.-A. (2013). An Improved Approach to the PageRank Problems. *Journal of Applied Mathematics*.