Advances and Applications in Mathematical Sciences
Volume 19, Issue 12, October 2020, Pages 1309-1321
© 2020 Mili Publications

STUDY AND ANALYSIS OF PENALTY BASED PAGERANK METHOD

AYUSH MISHRA[1], PANKAJ SHUKLA[2], PRATAP KUMAR[3], UTKARSH SHUKLA[4] and B. JAGANATHAN[5]

[1,2]School of Computing Science and Engineering
VIT University, Chennai 600127, India
E-mail: ayush.mishra2015@vit.ac.in
pankaj.shukla@vit.ac.in

[3]Galgotias College of Engineering and Technology
Plot No-1, Knowledge Park II
Greater Noida-201306, U.P., India
E-mail: pratap.kumar@galgotiacollege.edu

[2,5]Division of Mathematics
School of Advanced Sciences
VIT University, Chennai 600127, India
E-mail: utkarshshukla2912@gmail.com
jaganathan.b@vit.ac.in

Abstract

Everybody in this world is moving towards the age where they want to get the results as soon as possible with the highest efficiency. One such major example is of Google search engine in which people want to get the most appropriate results for their search. It has gone some decades that people are still working on page-rank algorithms for getting more and more efficient results. In this paper, we are introducing a new approach for finding the page-ranks which is called penalty based page-ranks algorithm. In this approach, we are considering the concept of penalty pages and nonpenalty pages which will be exemplified later in this paper. After checking the adequate working of the algorithm, we then are going to compare our results with the pre-existing algorithms such as Weighted Pagerank Algorithm, Simple Page-rank Algorithm because after comparative method only will prove a better tool to distinguish the differences among these various algorithms.

2010 Mathematics Subject Classification: 11D09, 11D99.
Keywords: continued fraction expansion, Diophantine equation, Gaussian integer solutions, Pell's equation.
Received November 20, 2019; Accepted January 17, 2020

Introduction

The internet retains a vast amount of data and information on almost any topic concerning any activity happening around the globe. Recently the access to Internet has become very convenient and easy but there are still numerous shortcomings when it comes to retrieving the information because of its randomness and vastness. In order to tackle this problem Search Engines came into existence which are software programmed to retrieve the information from the internet and display it to the user. Search Engines have been quite helpful in tackling the problem, but retrieval of highquality information is still an issue.

The most widely used algorithm for page ranking form retrieval of high quality information is the Google Page Rank, which has been the reason for its success and has made it the most popular search engine. The PageRank Algorithm was proposed by Larry Page and Sergey Brin in 1996 at Stanford University as a research project. In 1998, they published the First Paper on PageRank and later founded Google, which still uses PageRank algorithm as a basis for all its Search tools.

The algorithm takes into consideration, the hyperlinks to a webpage. The number of hyperlinks is a measure of the importance of page and hence higher the rank of the page. This very algorithm is computationally cheap and provides good results but it also ignores many factors such as penalty pages which play an important role in the rank of the pages. The proposed algorithm aims at more efficient rank calculations

- What is Web Page Ranking?

Web Page ranking aims at assigning ranking to the web pages on the net, based on their general importance, which includes factors like content, traffic etc. Various Search Engines use many Web Page Ranking algorithms to order the web pages whenever a search query is entered. Some of the popular Page Ranking algorithms include Page Rank, HITS etc.

- Why is Web Page Ranking required?

Whenever a user enters a query, he/she expects better results in the search engine. Web Page ranking ensures that, by showing the better ranked pages on the top, with better content quality and relevance. Also, with the constantly increasing size of the web, it becomes harder to find quality content. Page Ranking ensures that irrelevant pages which users rarely visit do not clutter the search result.

Existing Concepts:

Original Pagerank

This concept of webpage ranking was first established by Larry Page and Sergey Brin in 1998 when they were graduate students at Stanford University. This was one of the basic algorithm which changed the world and this revolutionary idea opened doors for many people to research in this field. All the algorithms which came after this were somehow based on the basic concept which Sergey and Larry had used. In this algorithm, it just makes the use of hyperlinks to a page for finding the relative ranking of several pages.

Larry and Sergey defined the whole World Wide Web as a graph $G = \{V, E\}$ in which V is a set of vertices or nodes which correspond to each page in the Web and E is the set of edges which correspond to hyperlinks among the pages. With the given information, they developed an adjacency matrix of size $n \times n$, where n is the total number of web pages, using the following formula:

$$1 i \neq j \wedge v_i \; points v$$

$$j \tag{1}$$

$$a_{ij} = \{0 otherwise\}.$$

They both came up with a formula in which they have used the above calculated adjacency matrix. The formula as proposed by them was:

$$PR_n = (1-d) + d * A(G) * PR_{n-1}. \tag{2}$$

In the above formula, d and $A(G)$ act as constants for a particular set of webpages and $PRn-1$ is the only variable in the formula they used. Here, d is the damping factor which lies in between 0 and 1. In a general scenario, it's value is taken as 0.85.

They performed several iterations until the iterations converged i.e. $PRn-1$ equals $PRn-1$ Therefore, on the basis of $PRn-1$ ranks of the given pages were determined.

Eigenvector Centrality:

This algorithm is based on the simple pagerank algorithm as discussed above but it has added a few more properties to the node by considering the weight of the hyperlinks, a node is connected to various other nodes. The concept behind this algorithm is that the page or node which is connected to higher valued nodes would have higher rank as compared to those nodes which are connected with low-valued nodes.

For implementation of this algorihtm, again let's consider a graph $G = \{V, E\}$ where V is the set of vertices and E is the set of hyperlinks. With the given information of G, an adjacency matrix is developed using the equation (1) given above. There is a difference between the adjacency matrix generated in Simple Pagerank Algorithm and in this algorithm and the difference is that, instead of filling 1 in the cells, weights of the hyperlinks have been filled. To calculate the centrality score, the following equation is used:

$$x_k = \sum_{t=1}^{n} a_{kt} x_t k = 1, 2, 3, \cdots, n. \tag{3}$$

The above equation is expressed in form of an eigen vector notation as show below:

$$Ax = \lambda x. \tag{4}$$

In aforementioned formula, the value of is the largest eigen value in accordance with the Perron-Frobenius theorem which states that, in a real square matrix having positive entries, the eigen vector corresponding to the unique largest eigen value consists of strictly positive components which is going to be very helpful in the given algorithm. Therefore, on the basis of finally obtained eigen vector, the rank of the pages are determined.

Weighted Pagerank:

Wenpu Xing and Ghorbani Ali researched in this field and came up with a new algorithm which is called Weighted [9] Pagerank Algorithm. In this algorithm, they added few more properties to the nodes i.e. webpages for getting more appropriate and efficient results. The importance of a particular webpage is calculated by finding the number of in-links and out-links to or from that particular page respectively. A webpage is given a relative rank on the basis of it's in and out weights using the below mentioned equations:

$$W_{(v,\, u)} = \frac{I_u}{\sum_{p \in r(v)} I_p}. \tag{5}$$

$$W^{out}_{(v, u)} = \frac{U_u}{\sum_{p \in r(v)} O_p}. \tag{6}$$

In this algorithm, initially each of the page is assigned a rank that equals 1 but after every iterations, the ranks are updated until the iterations converge. The iterative formula that is used in this algorithm is mentioned below:

$$PR(v) = (1 - d) + d^{*} \Sigma\, PR(v)^{*} W_{(v,\, u)} * W^{out}_{(v,\, u)}. \tag{7}$$

In the above mentioned formula, Win is the matrix which keeps the record of the weights of the inhyperlinks and W out keeps the record of the out-hyperlinks.

In equation (7), d is the damping factor which is going to always lie in between 0 and 1. But for the simplicity, they took $d = 0.85$. Teleportation probability [7] which we have used as d, is the possibility whether a web-surfer will continue it's surfing or will stop at a particular webpage.

Proposed Pagerank Algorithm:

The algorithm proposed by us involves a few basic keywords which are mandatory to be understood before proceeding further.

Penalty Pages

The pages which engage in practices which are against the Webmaster

Guidelines of Google, are penalized by Google and are called Penalty pages. This penalty results in the drop of the ranking of that particular webpage or website. In this algorithm, we are specifically going to focus on Link Based Penalty.

Stochastic Matrix:

A stochastic matrix is also referred as Transition matrix, Probability matrix or Markov matrix. The two basic properties of a stochastic matrix are as follows:

Let S (denoted by $(([Sij]))$ be an $n \times n$ matrix, then S is said to be a Stochastic Matrix only if

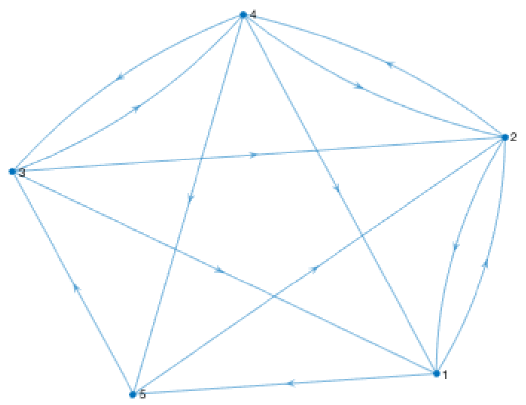
(1) All the elements are non negative real numbers; i.e.

For each i, $Sij \geq 0$

(2) $\Sigma\, ni = 1Sij = 1$.

The Sctochastic Matrix is classified into several types but in our proposed work, we are going to use Left Stochastic Matrx i.e. sum of elements of columns is equals to 1. Another useful property of stochastic matrix which we are going to use in our proposed work is that the largest possible eigen value of the matrix is 1 and all other eigen values are positive but less than 1.

After going through the basic keywords that we are going to use frequently, we are moving towards the basic idea behind the proposed algorithm. There are many pages/sites on the web that don't follow certain guidelines or use Black Hat SEO techniques. Therefore, it's necessary to consider those pages while finding the rank of the pages because these penalty pages are going to effect the pages from which or to which they are connected.



Now we are going to discuss the algorithm which we have proposed in this paper step by step. Initially we are given with a graph $G = \{V, E\}$ where V is the set of vertices (i.e. pages) and E is the set of edges (i.e. hyperlinks). And we know that maximum nodes are non-penalty pages but few are penalty pages therefore we are going to make an adjacency matrix $A(G)$ as generated in existent algorithm but with few major changes. The difference that we are going to introduce is that instead of 1, we will use 0.85 or 0.15 based on the hyperlink where it is pointing to. If the hyperlink is pointing towards non-penalty pages, then we will assign 0.85 in the matrix but if it is pointing towards a penalty page, then it is assigned as 0.15. The adjacency matrix that is developed is based upon the assumption that the j^{th} node is pointing towards i^{th} node in a matrix $A([i, j])$. Now we will get our resultant matrix $A(G)$ by following two steps:

First an adjacency matrix is generated on the basis whether a hyperlink exists from node Vj to Vi, and if it exists, then it is assigned the value 1.

$$A([i, j]) = \{1 \, \textit{if} \ i! = j \text{ and } Vj \text{ points to } Vi$$

$$\{0 \ \textit{otherwise}. \tag{8}$$

In the next step, wherever the value is 1, we start checking whether any node Vj pointing towards a penalty page or not. If Vj is pointing towards a penalty page Vi, then it's value is reassigned as 0.15 from 1 else but if it's pointing towards a normal page i.e. non-penalty page, then the value is changed to 0.85 from 1 because the concept says that the users should be prevented to reach the penalty pages that is why the hyperlinks pointing towards penalty pages are assigned very low weights as compared to normal pages.

$$A'[i, j] = \{0 if\ A[i, j] = 0$$

$$\{0.85\, if\ Vi \text{ is non-penalty page}$$

$$\{0.15\, if\ Vi \text{ is non-penalty page.} \tag{9}$$

After getting the updated adjacency matrix $A'(i, j)$, next step is to normalize the updated adjacency matrix A' coulmn wise. For normalizing the matrix, coulmn wise, summation (Sj) of all the elements of a particular column is taken and then divide each element of that column by the value Sj for normalization.

$$A[i, j] = A[i, j]/Sj \tag{10}$$

0	0.5000	0.4595	0.4250	0
0.5000	0	0.0811	0.0750	0.1500
0	0	0	0.4250	0.8500
0	0.5000	0.4595	0	0
0.5000	0	0	0.0750	0

The normalization step that is done above, results in a Left Stochastic Matrix which is going to be very helpful in extracting few results on the basis of it's properties. Since there is a property of Stochastic Matrix that it has maximum possible eigen value of $A'(i, j)$ as 1 and all other eigen values will lie in between 0 and 1. Therefore in the proposed algorithm, the eigen vector corresponding to this maximum eigen value i.e. 1, is taken into consideration for finding the ranking of each webpage.

Algorithm

• Take input from the user for the number of nodes in the graph in "nodes"

• Create an array of length "nodes" with values from 1 to nodes "non-penalty"

• Take input from the user for number of penalty pages in the graph "penalty-count"

• Create an empty array "penalty"

• Loop from 1 to penalty-count

o Select a random value from non_penalty array

o Add that value to the array penalty

o Remove the value from the non_penalty array

• "penalty" contains random nodes equal to "penalty_count" and those nodes are not in the array "non_penalty"

• create two array arr1 with values 0.15 and 0, and the array arr2 with values 0.85 and 0

• create a 20 cross 20 zeros matrix "matrix"

• Loop in "*i*" from 1 to the length of "penalty" pages

o Initialize an empty array "n"

o Loop from 1 to "nodes"

• Add a random value to n selected from arr1

o Initialize "row_number" with value of "penalty" at "i"

o Replace the "row_number" in "matrix" with n

• Loop in "*i*" from 1 to the length of "non_penalty" pages

o Initialize an empty array "*n*"

o Loop from 1 to "nodes"

• Add a random value to n selected from arr2

o Initialize "row_number" with value of "non_penalty" at "*i*"

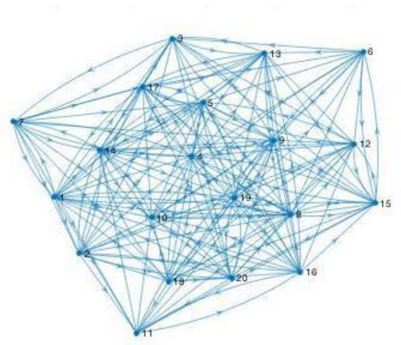
o Replace the "row_number" in "matrix" with *n*

• Set the main diagonal of "matrix" to zero to remove self link

• Calculate the sum of each column of the matrix and divide the values in the column with respective sum to make them stochastic

• The rank of the pages is the eigen values corresponding the eigen vector with value equal to one.

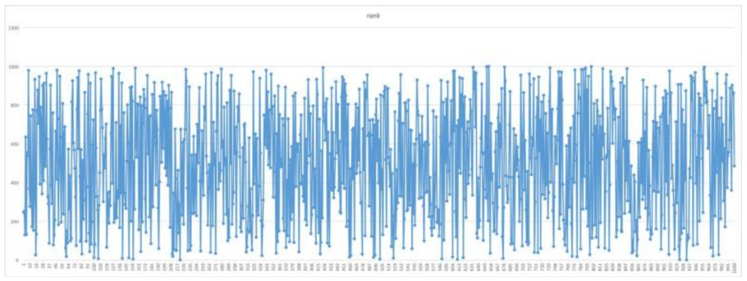
Experimental Analysis and Results for Synthetic Data Field



Nodes	Rank of the pages	Page rank values	Rank of the pages	Eigen Page Rank	Rank of the pages	Proposed page rank
1	5	1.2044	5	0.2682	6	0.2465
2	10	1.0143	11	0.2137	15	0.1831
3	18	0.7739	17	0.1648	20	0.0277
4	17	0.7912	19	0.1471	1	0.4016
5	6	1.183	3	0.2731	12	0.2188
6	12	0.9462	12	0.2123	19	0.0289
7	11	1.0072	8	0.2302	13	0.2124
8	3	1.2273	2	0.2797	4	0.2792
9	9	1.0491	10	0.2194	7	0.2356
10	4	1.223	4	0.2728	11	0.2234
11	20	0.4645	20	0.0918	5	0.2486
12	7	1.1227	6	0.2652	14	0.1844
13	19	0.7729	16	0.179	16	0.0542
14	2	1.2469	7	0.2577	3	0.3106
15	16	0.8548	18	0.1537	2	0.3322
16	15	0.8812	15	0.1957	10	0.2249
17	13	0.9365	14	0.2056	17	0.0528
18	1	1.3051	1	0.2879	8	0.2323
19	8	1.0671	9	0.2302	9	0.2271
20	14	0.9288	13	0.2072	18	0.0431

The formative of table reveals comparative values among the simple page rank algorithm, the eigen value centrality algorithm and the proposed algorithm. The above graph the nodes 3, 6, 13, 17 and 20 are penalty algorithm (defined by us). It is evident that the penalty pages are given the least rank by the proposed algorithm ranks of 3, 6, 13, 17, 20 are 20, 19, 16, 17, 18 respectively. The eigen value centrality and the simple based rank don’t take into consideration the penalty pages and don’t allot the penalty pages the lowest values but treat them like normal nodes.

Analysis For 1000 Nodes



Total number of nodes into consideration are 1000. Total number of penalty pages are fifteen: 127, 158, 279, 422, 633, 652, 655, 677, 741, 791, 799, 815, 849, 957, 958 as (defined by us). The main objective of our algorithm is to consider the penalty pages and provide them with lowest rank. The rank allotted to the pages are 990, 991, 987, 994, 995, 998, 1000, 997, 993, 992, 999, 988, 989, 986 and 995

Conclusions

As it is evident from the above value comparison that the ranking obtained from the proposed algorithm is somewhat different from the values obtained from the eigen value centrality and simple page rank algorithm because they don’t take into consideration the penalty pages unlike the proposed algorithm. Hence, one can find that the penalty pages tend to receive lower ranks as compared to non penalty pages. Moreover, one can also find that the proposed algorithm is less computationally expensive when compared to the other page ranking algorithms like simple pagerank and eigen value centrality. The complexity of the proposed page rank is $O(n^2)$.

References

[1] B. Jaganathan and D. Kalyani, Category Based Page Rank Algorithm, International Journal of Pure and Applied Mathematics, 2015 Aug; 101(5):811-820.

[2] B. Jaganathan and D. Kalyani, Penalty-Based Page Rank Algorithm, ARPN Journal of Engineering and Applied Sciences 10(5) (2015), 2000-2003.

[3] B. Jaganathan and D. Kalyani, Weighted Page Rank Algorithm Based on In-Out Weight of Webpages, International Journal of Science and Technology 8(34) (2015), 1-6.

[4] S. D. Kamvar, M. T. Schlosser, H. Garcia-Molina, The eigentrust algorithm for reputation management in p2p networks, 2015 July p. 640-651.

[5] Krishnan, Vijay Raj and Rashmi, Web Spam Detection with Anti-Trust Rank, (PDF). Stanford University. 2015 Jan p. 1-5.

[6] L. Page, S. Brin, R. Motwani and T. Winograd, The page rank citation ranking: Bringing order to the Web, Technical report, Stanford Digital Library Technologies Project; 1998 Jan. p. 1-17.

[7] R. Lambiotte and M. Rosvall, Ranking and clustering of nodes in networks with smart teleportation, 2 June, 2012.

[8] W. Yu, X. Lin, W. Zhang, L. Chang and J. Pei, More is Simpler: Effectively and Efficiently Assessing Node-Pair Similarities Based on Hyperlinks, In VLDB 13. Proceedings of the 39th International Conference on Very Large Data Bases, 2013 p.13-24.

[9] W. Xing and A. Ghorbani, Weighted page rank algorithms, Proceedings of the Second Annual Conference on Communication Networks and Services Research (CNSR'04); IEEE. 2004 May 19-21. p. 305-314.

Appendices

```
nodes<- Input number of nodes
non_penalty = 1:nodes
penalty_count<- Input number of penalty pages
penalty<- zeros(penalty_count)
for i = 1:penalty_count
n =random(non_penalty)
penalty[i] = n
non_penalty.pop(n)
end
arr1 = [0.15,0]
arr2 = [0.85,0]
matrix = zeors(nodes)
for i = 1:penalty_count
n = zeros(1,nodes)
for j = 1:nodes
n(j) = random(arr1)
end
row_number = penalty_pages(i)
matrix(row_number,:) = n
end
for i = 1:length(non_penalty)
n = zeros(1,nodes)
for j = 1:nodes
```

```
n(j) = random(arr2)
end
row_number = non_penalty(i)
matrix(row_number,:) = n
end
for i=1:nodes
matrix(i,i) = 0;
end
column_sum = sum(matrix,1)
for i = 1:nodes % making the array stochastic
column = matrix(:,i);
If column_sum(i) ~= 0
column = column/column_sum(i);
end
matrix(:,i) = column;
end
[V,D]= eig(matrix);
D = real(D);
V = real(V);
Z = round(D,5);
val = find(Z==1);
disp(val)
i_comp = ceil(val/nodes);
j_comp = val - ((i_comp-1)*nodes);
required = abs(V(:,j_comp));
    disp(required)
```