The PageRank Citation Ranking: Bringing Order to the Web

*CS333 Algorithm Analysis Project Report

1st Alkın Korkut
Computer Science
Ozyegin University
Cekmekoy, Istanbul
alkin.korkut@ozu.edu.tr

2nd Emir Mehmet Eryılmaz
Computer Science
Ozyegin University
Cekmekoy, Istanbul
emir.eryilmaz@ozu.edu.tr

3rd Ezgi Nur Alisan
Computer Science
Ozyegin University
Cekmekoy, Istanbul
ezgi.alisan@ozu.edu.tr

***Abstract*—As the internet develops rapidly, many new websites are added to the web every day. In terms of the optimization and usability of search engines, serving the right website, which the user wants to reach from billions of sites as efficiently as possible, has gained importance. The idea of content-based search has been put forward to solve this problem, however this solution is vulnerable to keyword manipulation and spamming. Therefore, the PageRank algorithm, which is a global ranking of all web pages, based solely on their location in the Web's graph structure has been introduced as a more optimal solution.**

I. Introduction

With the rapid expansion of the World Wide Web, accessing information has become more complex. Searching on the Web has become more and more difficult for users to find relevant and useful information on web pages on the Internet due to the existence of a great number of web pages and the diversity of them. An approach used by traditional search engines of those times was displaying search results based on how many times a particular search term appeared on a webpage. This technique was generally open to manipulation. In addition, it was not very good at finding exactly the relevant content that users were trying to reach. Another approach was applying standard citation analysis techniques which determines the importance of a web page by counting the number of pages (citations) that point to it. However, this approach disregards the importance of the pages which point to a specific page. For example, in the context of journalism, a backlink (reference) by The Guardian is more important than a backlink by a local newspaper website. As these proposed algorithms do not exactly meet the needs of the users due to the problems addressed previously, Page and Brin introduced another approach called PageRank.

A. PageRank

PageRank is an algorithm which determines the importance of web pages by analyzing the links that connect them. It is used to evaluate web pages in an unbiased and automated manner, which can accurately reflect the level of interest and attention shown by humans towards those pages. With this algorithm, a rank for each web page in the entire web graph is calculated. The rank of a website is directly proportional to the importance of that website. While determining the page order, a solution was presented by focusing on the number of other pages referring to that page and the importance of these pages. Other pages that refer to that page are called backlinks (inedges), and the pages to which the page refers are called forward links (outedges).

II. Background

The World Wide Web is represented by graph structure. The nodes of the graph represent the web pages and the edges represent hyperlinks between the pages. Hyperlinks consist of backlinks and forward links. Each node has a page rank associated with its backlinks. The PageRank of the webpage is computed by summing all the weights of its backlinks which indicates the importance of the link. Weight of the edges are calculated by dividing the rank of a node by the number of its forward links.

$$R(u) = c \sum_{v \in B_u} \frac{R(v)}{N_v} \tag{1}$$

This formula represents the calculation of the PageRank. The ranking function R(u) shows the rank of the page "u". Node v is used to describe a node that has a forward link to node u. Nv specifies the number of forward links that node v has. B(u) includes all the backlinks which the node u has. "c" is the factor used for normalization. This equation can be solved by starting with some nodes and doing the calculation recursively until it stops changing. Simple demonstration of the algorithm can be seen in Figure 1.

The equation can be stated as a matrix formula.

$$R = cAR \tag{2}$$

A is a square matrix in which its rows and columns represent the nodes. The value of A(u,v) is 0 if there is no edge between u and v, and 1/Nu if there is an edge. R is a vector that consists of the ranks of all nodes. According to the equation above, R can be deduced as the eigenvector of A, and c can be deduced as the eigenvalue of A.

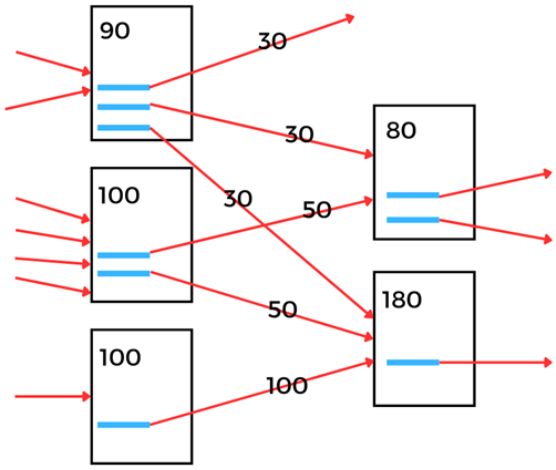

Fig. 1. Pages and their links

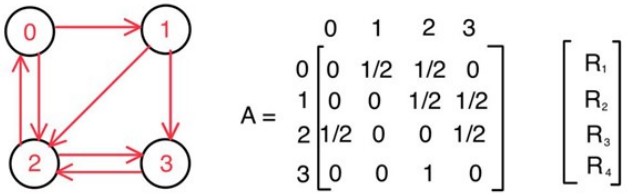

Fig. 2. Example of Structure

An example of a graph can be seen in Figure 2. This graph contains 4 web pages with numbers 0,1,2,3 respectively. An arrow that leaves from node u that points to node v represents forward link from u to v. For example, there is an arrow that leaves from node 1 and points to node 3 in Figure 2. This represents a forward link of node 1 that points to node 3. This edge can also be seen as a backlink of node 3 which comes from node 1. By using this graph example, the matrix A and the vector R can be constructed. Since the graph has 4 nodes, A matrix will be a 4x4 matrix and the R is a 4x1 vector. In this calculation the multiplication of each row v in the transpose of A square matrix with the Rank vector equals to the PageRank of the node v. In this calculation we assume that the web surfer always follows the outlinks, however this is not necessarily the case in web surfer. A surfer might also jump to a random web page that doesn't have a backlink from the webpage that the surfer is currently on. To take this situation into account we introduce a constant called the damping factor(c) that represents the probability of the surfer following the outlinks. Based on this, the probability of the surfer jumping to a random webpage can be calculated as 1-c. The damping factor is usually chosen as 0.85.

$$R(u) = c \sum_{v \in B_u} \frac{R(v)}{N_v} + (1-c)E(u) \tag{3}$$

The updated formula can be seen above. Here E is an Nx1 vector where N corresponds to the number of web pages in the graph. The value of E(u) corresponds to the possibility of the surfer randomly jumping to the web page u. There is one issue about the links. There are links that point to pages with no outgoing links. This is a problem because it is not clear where their ranks should be distributed. These are usually the result of broken URLs, removed content, or pages that have not been downloaded yet. Since dangling links have no effect on the ranking of any other page directly, these links can be removed until the PageRanks converge. Then they can be added back to the graph.

III. Main Section

A. PageRank Explanation

In the PageRank implementation, Brin and Page (1998) determined to assign unique integers to the URL's for storing the hyperlinks. These integers are used to calculate the PageRank values of the pages until they converge, using the PageRank formula mentioned in the previous section (see section 2). Firstly, the dangling links are removed from the graph. The initial assignment for the ranks must be made. It does not affect the final values much, but the rate of convergence can be affected. Convergence criteria is met when the difference between the current pagerank values and the pagerank values that were calculated in the previous iteration is smaller than an epsilon value. PageRank can mostly be used in search problems. There are other algorithms used in search problems besides PageRank. By making comparisons in the article, it has been observed that the PageRank algorithm is more applicable than other existing solutions. It is also more trustworthy since if there is a forward link from an authoritative page (highly ranked) to the current page, it will be more likely to be trustworthy. For example, if a page has a backlink from a well-known news site, it will be more reliable than a backlink from the local news site. A crucial element in the PageRank calculation is vector E, which represents the distribution of rank across web pages. We have already mentioned in the previous section that the value of E(u) corresponds to the possibility of the surfer randomly jumping to the web page u. It can either be uniformly distributed such that every page has the same probability of being jumped to or it can be personalized where the E matrix is given as input. The E matrix introduces a bias towards the user's preferred pages, making them more likely to receive higher rankings in the personalized ranking order. Personalized PageRank is used when there is a desire to direct a user to specific websites based on their preferences or interests instead of uniformly distributed situation. These page ranks are difficult to manipulate by commercial interests. However, they may be subject to some manipulations from time to time.

B. PageRank Implementation

Based on the mathematical background for Pagerank algorithm, we implemented a basic program that calculates pagerank values of the nodes on the graph. Source code of the program can be seen in the Appendix part. The program

that we developed has only one function whose name is calculatePagerank. This function calculates the pagerank of the graphs. It takes four parameters which are links, dampingFactor, epsilon, maxIterations. Links parameter refers to an array that has forward links of all nodes in the graph. Damping factor is taken as 0.85 by default. We decided to choose epsilon value as 1e-8 that determines the point at which the algorithm should converge. If the algorithm does not converge it will stop at the iteration which is determined by maxIterations parameter. The inital pageranks of the nodes are assigned uniformly as 1/numberOfNodes value. Then a square matrix A is constructed. A is a NumberofnodesXnumberofnodes matrix where each row and column represent a node of the graph. If there is an edge from node i to node j, A[i][j] will be assigned to 1/(numberof forwardlinks from i), if there is no edge, it will be assigned to zero. E is an Nx1 vector where N corresponds to the number of web pages in the graph. The value of E(u) corresponds to the possibility of the surfer randomly jumping to the web page u. We have determined each element of E uniformly as 1/N but it can also be taken as a paramater if personalization is desired. In each iteration of the for loop the pagerank is updated according to the formula mentioned in Equation 3. Difference between the new ranks and old ranks is calculated. Then, if the absolute value of difference is smaller than epsilon value we break out of the for loop. At the end, the function terminates by returning final pageranks of the nodes.

IV. Discussion

The article deals with several different issues in terms of the time complexity of the PageRank algorithm. They mentioned that memory is allocated for the weights for every page by using a big database. As we did not have a huge amount of data for the measurements, it was not necessary for us to use a database. Since the database is sorted initially, accessing the data on disk has a linear time complexity which is $O(N)$. In terms of convergency, they observed that when the size of links doubles, the number of iterations in which the algorithm converged changed slightly. There is no definitive study in the area of algorithm complexity in the article. Therefore, we focused on examining the complexity of our implementation. We do square matrix and vector multiplication in each iteration of the algorithm which takes $O(N^2)$ time complexity. Since it iterates over the max iterations variable, the overall complexity will be $O(N^2 * maxIteration)$. However, it can mostly converge before reaching the max iteration value, because of this the algorithm will usually perform better.

V. Conclusion

As a result, the PageRank algorithm, which was created based on graphics and matrix subjects, is intended to be used by search engines to bring relevant results for what the user is looking for. It achieves this by using a ranking system that assigns a rank value to every web page, considering their position within the web graph structure. It does not take their content into consideration. Compared to other search engine algorithms, PageRank has been found to be more useful. In this context, to examine the working mechanism of PageRank, we implemented a program that simulates PageRank, and we analyzed its performance.

References

[1] Page, L., Brin, S., Motwani, R. and Winograd, T. The PageRank Citation Ranking: Bringing Order to the Web. Stanford Digital Library Technologies Project, 1998.

[2] GeeksforGeeks. https://www.geeksforgeeks.org/page-rank-algorithm-implementation/

[3] GeeksforGeeks. https://www.geeksforgeeks.org/ranking-google-search-works/

[4] GeeksforGeeks. https://www.geeksforgeeks.org/google-search-works/

[5] Wikipedia. https://en.wikipedia.org/wiki/PageRank

VI. Appendix

```python
import numpy as np

def calculate_pagerank(links, damping_factor=0.85, epsilon=1e-8,
max_iterations=100):
    num_pages = len(links)
    initial_rank = 1.0 / num_pages
    ranks = np.full((num_pages,1),initial_rank) # We initialize PageRanks with
1/ num_pages
    A = np.zeros((len(links),len(links)))
    for i in range(len(links)):
        for j in range(len(links[i])):
            A[i][links[i][j]] += ((1/len(links[i])))
    for _ in range(max_iterations):
        new_ranks = ((1 - damping_factor) / num_pages) + damping_factor *
np.matmul(A.T, ranks)
        diff = np.sum(np.abs(new_ranks - ranks))
        ranks = new_ranks
        if diff < epsilon:
            break
    return ranks
```

Fig. 3. Our Implementation