Fast Ranking System for Proposed Page Rank Algorithm using Markov Chain

Khaing Thanda Swe
CEIT, Mandalay Technical University
khaingthandasweutycc@gmail.com

Htet Aung Kyaw
CEIT, Mandalay Technical University

Abstract

Information retrieval on the web is significant and furthermore complex activity for web mining. Because of enormously increased the number of websites on the Internet, the execution of PageRank Algorithm should be easy and faster in operation. So the system proposes the fast execution time in PageRank with the reduction of iteration step rather than the original PageRank Algorithm. The proposed system contains two main parts. The first part is to create simple website using mostly HTML, CSS and also bootstrap, front-end framework for web design in the demonstration of the proposed system. The second part is to make the line number extractor using simple web crawler to get the required information for proposed PageRank Algorithm. Then, the PageRank algorithm is executed by using the data from web crawler. Finally, the convergence value is taken to specify the iteration count during the operation of PageRank Algorithm. The main point of proposed system is faster execution time than the original PageRank Algorithm with reduction in iteration steps.

Keywords: PageRank, HTML and CSS

1. Introduction

The web is huge and very popular source of information which provides quick response to the users and also it can reduce the physical movements for the users. The web data are noisy that become from two major sources. Firstly, the web may contain many pieces of information, for example, main content, routing links, advertisements, copyright notices, privacy policies, and so on. The second one is due to the fact that the web does not have quality control of information, that is, one can write almost anything that one likes. Large amount of information on the web is low quality, erroneous or even miss-leading. Therefore, retrieving required web page on the www, effectively and efficiently, becomes very difficult.

Web crawlers are confronted with various troublesome issues in keeping up or improving the quality of their performance. These problems are either unique to this domain, or novel variants of problems that have been studied in the literature [1]. The goal of this proposed system is to raise awareness of several problems that could benefit from increased study by the research community. Interesting and difficult problems are deliberately ignored that are already the subject of active research.

The proposed system is to reduce the execution time of PageRank Algorithm. In order to apply the PageRank Algorithm, firstly it needs to learn the working flow of original PageRank Algorithm. This algorithm is depended on the output of crawler which can extract the entire website. The crawlers collect the data, which are in-links, out-links, total number of web pages and also popularity of web site. Then, the PageRank Algorithm is operated by using the data from the crawler's output.

2. Literature Reviews

PageRank Algorithm was originally developed by Sergey Brin and Larry page as a Ph.D thesis at Stanford University and it was original of Google Search Engine. The first advantage is that can compute the rank of web page at crawling time, so it responses quickly to user query. The second advantage is less susceptible to localized links as it uses the entire web graph to generate page ranks rather than a small subset. The weak point for this algorithm is that it leads to spider traps if a group of pages has no outgoing links to another internal or external group of pages. Then, dead ends and circular references will reduce the front page's PageRank [2].

Weighted PageRank Algorithm was an extended PageRank Algorithm which was developed by Wenpu Xing and Ali Ghorbani. As advantages, it performs at crawl time rather than query time, hence it has a higher efficiency and the rank value of a page is not equally divided amongst outgoing links rather it is assigned according to the important of each page it is linked to. The disadvantage is that turns less relevant pages to a user query as the ranking is based on web structure and not content. Moreover, it is a static algorithm that

means pages that are popular tend to remain popular throughout which does not guarantee the desired information to user query [3].

HITS Algorithm was developed by Jon Kleinberg. In this search, the web pages are divided into two types of pages. There are Hubs and Authorities pages. Centers are pages that go about as an asset list, containing a decent wellspring of connections. The good thing for HITS Algorithm has the ability to rank pages according to the query string, resulting in relevant authority and hub pages. And then, the important pages are obtained on basis of calculated authority and hubs value. There are four disadvantages in HITS Algorithm. The very first one is mutual relationship between hubs and authorities that can lead to erroneous weights. The second fact cannot easily identify whether a page is hub or authority. The third one is topic drift. It may not produce relevant document as the algorithm weighs all pages equally. Finally, it is not efficient in real time [3].

In paper of Brin and Page, they present Google, a prototype of a large-scale hyper textual web search engine which makes heavy use of the structure present in hypertext. The point of this paper is how to build a practical large-scale system which can exploit the addition information present in hypertext. The Google search engine has two important features that it helps to produce high precision results. First, it makes use of the link structure of the Web to calculate a quality ranking for each Web page. This ranking is called PageRank [4].

In Atul Kumar Srivastava paper, it has given an experimental analysis of PageRank computation for different value of the damping factor. It has observed that for value of d=0.7, PageRank method takes fewer numbers of iterations to converge than d=0.85, and for these values of the top 25 web pages returned by PageRank strategy in the SERP are practically same, just some of them trade their positions. From the trial results it is seen that estimation of damping factor d=0.7 takes inexact 25-30% less quantities of cycles than d=0.85 to get intently indistinguishable website pages in top 25 outcome pages for customized web search, particular creeping, intra-web internet searcher [4].

From these results, the search engine, especially Google, decide how important this web site is by using many algorithms. Among them, this proposed system focuses on PageRank Algorithm which can calculate the ranking value of web site in pre-calculated result.

3. Proposed System's Methodology

Before calculating page rank value of web site, the algorithm needs the information of specific website by using web crawlers. Web crawlers are program to extract the entire web site and download required information for PageRank algorithm. Therefore, the crawlers play an important role in the calculation of page rank. Background theory of web crawler is based on the Markov Chain process [5]. This section is discussed about Markov Chain techniques that are based on the crawler's execution.

3.1. Markov's Chain

A row-stochastic matrix is a non-negative n×n matrix where each row sums to one. It is important here for it use in Markov chains[05Jia]. A Markov chain is a stochastic process that satisfies the Markov property, a memoryless property where the probability independent of past behavior. The Internet network matrix can also be defined by using the transition probability:

$$S_{ij}=P\left(X_t=P_j \middle| X_{t-1}=P_i\right) \quad (1)$$

With a transition matrix **S** then can find a probability distribution vector **p,** a non-negative vector where:

$$p^T=(p_1 p_2 p_3 \dots p_n) \quad (2)$$

The probability distribution vector in Markov Chain is:

$$p^T(k)=\left(p_1(k)p_2(k)\dots p_n(k)\right), k=1,2,3,\dots, \text{where } p_j(k)=p\left(X_k=S_j\right) \quad (3)$$

The initial distribution vector:

$$p^T(0)=\left(p_1(0)p_2(0)\dots p_n(0)\right), \text{where } p_j(0)=p\left(X_0=S_j\right) \quad (4)$$

The state space of the model depends on the current state and not on the sequence of events that preceded it. This is also known as the Markov property. Markov model can have N^{th} order:

- 0^{th} order models depend on no prior state.
- 1^{st} order models depend on one previous state.
- n^{th} order models depend on n-1 previous states.

Markov property:

-Behavior at time t depends only on its state at time t-1.

-Sequence of outputs produced by a Markov model is called a Markov chain.

-Process of Markov chain model in order to perform page Ranking [5].

3.2. Transition Matrix

Transition Matrix T is n × n matrix where n represents the number of states [15Xia]. The matrix is formed from transition probability of Markov process. Transition matrix t_{uv} is equal to the probability of moving from state v to u at time t in each entry. Therefore, $0 <= t_{uv} <= 1$ must be true for all u and v [6].

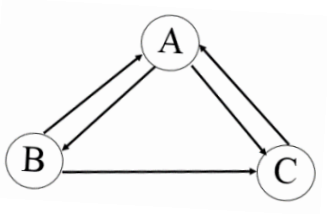

Figure 1. Simple Web Graph

In Figure 1, there are three pages {A, B, C}. The transition probability of this graph is:

$$t_{uv} = \begin{bmatrix} 0 & .5 & 1 \\ .5 & 0 & 0 \\ .5 & .5 & 0 \end{bmatrix} \quad (5)$$

This matrix determines the probability of making a transition from one state to another. There are three states A, B and C. Page A reaches page B and page C with probability of 0.5 each. From page B it reaches page A and C with 0.5 probability and page C reaches page A and B with 0.5 each.

3.3. Page Rank Algorithm

Google uses PageRank Algorithm[4] in search engine. And this algorithm is also heart of Google. The function of PageRank Algorithm is to decide how important the web page is. Brin and Page gave the algorithm for finding the page rank as:

$$PR(A) = (1-d) + d\left[\frac{PR(T_1)}{L(T_1)} + \ldots + \frac{PR(T_n)}{L(T_n)}\right] \quad (6)$$

Brin and Page also claimed that the PageRanks formed a probability distribution over all web pages, so that the sum of all of them would be 1. This is not the case with the algorithm they gave, so they modified as below:

$$PR(A) = \frac{(1-d)}{N} + d\left[\frac{PR(T_1)}{L(T_1)} + \ldots + \frac{PR(T_n)}{L(T_n)}\right] \quad (7)$$

Where N is the total number of pages in the network. With this modification it then forms a probability distribution. By performing a number of iterations of the algorithm, the PageRank of all pages in the network can be determined [7]. The modified one can be written generally as:

$$PR(a_i) = \frac{(1-d)}{N} + d \sum_{a_j \in G(a_i)} \frac{PR(a_j)}{L(a_j)} \quad (3.11)$$

Where, a_i = Web Page

a_j = A page with an outgoing link to a_i.

The link matrix can be defined by:

$$S_{ij} = \begin{cases} 1/l_{pj}, & \text{if } P_j \text{ link to } P_i \\ 0, & \text{otherwise} \end{cases} \quad (8)$$

4. Design and Implementation Phase

The purpose of proposed method is to reduce the execution time in PageRank Algorithm. The original algorithm was looped through between 50 and 100 times in iteration and stopped when the results were convergent. But for this proposed system it is firstly considered about convergence of results then iterated in exact iteration times as shown in the following system design.

The proposed system needs to put the input URLs which can crawl by crawler. The crawler in this method produced the number of links which are attached in web site. Because the PageRank algorithm needs that results in order to weigh for that web site. After crawling the web site, the proposed method operates in a specific iteration count execution.

The measurement for proposed method is discussed by using two web sites which have different in total number of web pages in structure. The results are taken average in the execution time.

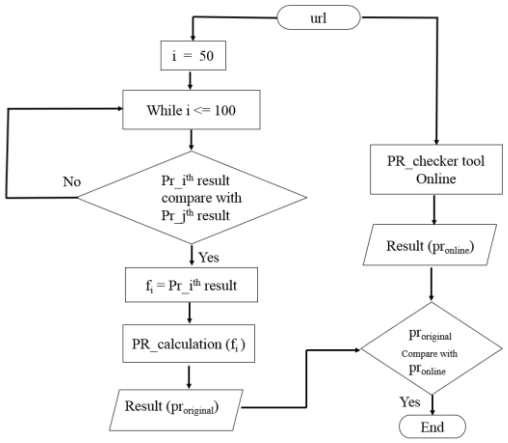

Figure 2. System Design for Proposed PageRank Algorithm

Then, this system produces one page-rank value for the highest web page. In order to validate that result is valid or not, a tool is taken from the Internet which can check the page rank of web site, web master tool. If the two results are equal, it can be concluded that the calculation for PageRank Algorithm is valid.

This proposed method needs to check stability which depends or not on the total number of web pages in specific web site. Therefore, to verify the stability of the method, two web sites are tested which are different in total number of web pages. The first web site includes five web pages and the second one occupies eleven web pages.

The original pseudo code is used to compare the proposed method. The pseudo code for original Google PageRank Algorithm is as follows:

4.1. Proposed Pseudo for PageRank(G)

Input: Let G represent set of nodes or web pages
Output: An n-element array of PR which represent PageRank for each web page
while i← 0 to n-1 do
Let A be an array of n elements
A [i] ← 1/n
d ← some values 0<d<1, e.g.0.15,0.85
Repeat
while i← 0 to n-1
do
Let PR be a n-element of array
PR [i] ← 1-d
For all page Q such that Q links to PR[i] do
Let O_n be the number of outgoing edges of Q
PR [i] ← PR[i] + d * A[Q]/O_n
If the difference between J and PR is small
do
Return PR
while i← 0 to n-1 do
A[i] ← PR[i]

The original program follows the above pseudo code using "for loop". The proposed method used "while loop" rather than "for loop". And then, the proposed one splits the original into two operations. The first one is the iteration specific count value then using that value PageRank Algorithm is operated instead of looping 50 and 100 times. Because Google iterate in looping is just between and 50 and 100 times.

This proposed method needs to check stability which depends or not on the total number of web pages in specific web site.

5. Comparison and Recommendation

As an implementation programming language, Python 3.7 is used in this system. The proposed system is focused on the execution time of pre-calculated ranking value of web sites. Therefore, the results are taken in average 10 times and compared with the original and proposed gaps. As a result, the following figure represents the comparison of original and proposed method in microseconds and takes ten times average for both.

The two hosted websites are named into "fortwoworld" and "webservices" websites that are presented in Figure 3 too. Using original method, a website which had five pages, it took time 6776.676 milliseconds in execution but in proposed, it took 7.004 milliseconds. The next website which has eleven webpages in original, it took 166408.811 milliseconds but in proposed, it took 1.011 milliseconds. These results are based on the ten times in average and damping factor value is 0.995, for both websites.

In the proposed system, it changes the various damping factor up to three such as (d=0.995, d=0.85 and d=0.3). Finally, by changing damping factor values, the proposed method can be stable because it does not change in the execution time of PageRank calculation.

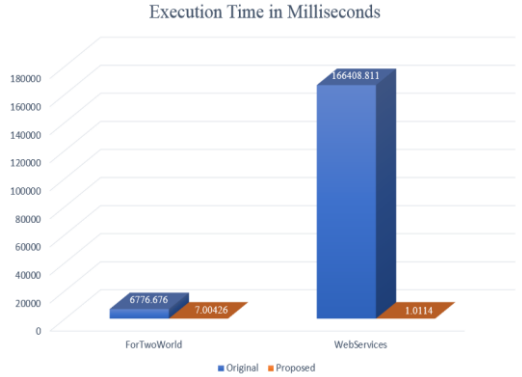

Figure 3. Average Result for Proposed Method in Milliseconds (d = 0.995)

Figure 4. Average Result for Proposed Method in Milliseconds (d = 0.85)

Damping factor value is also important in the calculation of PageRank value of web site. Therefore, the proposed system needs to test by changing various damping values. As the smallest damping value, it chose 0.3 and the results are shown in the following figures.

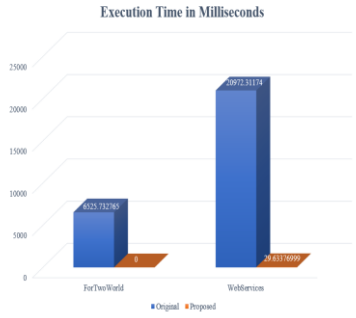

Figure 5. Average Result for Proposed Method in Milliseconds (d = 0.3)

Therefore, the proposed method can reduce the execution time obviously than the original. It may be better performance for search engine in the calculation of pre-calculated ranking value. And then, the proposed system does not depend on the total number of webpages.

6. Conclusion

In this paper, it is studied the pre-calculated PageRank value of original Google PageRank Algorithm in program execution. First, the crawlers are extracted the total in-links, out-links and numbers of web page by using link extracting program which was developed in Python programming language. Then, it kept those results and fed as an input to another function, proposed one, which calculated the specific iteration count value. Finally, the algorithm used link number and specific value, in iteration count, to calculate the pre-calculated PageRank value. Because of dividing the original into two operations, the execution time is faster than the original.

For more specific in search query keywords, it can depend on the density of the keywords. As further extension, it is based on the keyword density and hit rate from the user query then calculated the PageRank of web site.

References

[1] Tiancheng,L et al;Yuchen,Q; Xi,C.andXiobai,S.: Damping Effect on PageRank Distribution, arXiv:1806.00127vl, 31 May 2018.

[2] Atul,K.S; Rakhi,G andMishra,P.K.:Discussion on Damping Factor Value in PageRank Computation, MECS, DOI:105815, September 2017.

[3]Taruna,K;Ashlesha,G.andAshutosh,D.:Comparative Study of PageRank and Weighted PageRank Algorithm, International Journall of Innovative Research in Computer and Communication Engineering, Vol.2, Issue 2, February 2014.

[4] Sergey,B.andLawerence,P.: The Anatomy of Large-Scale Hypertexual Web Search Engine, Computer Science Department, Stanford, CA 94305, USA, 1998.

[5] Chaitra,L.M.andAyesha,A.M.:Pre-Processing and Analysis of Web Server Logs, International Journal of Innovative Research in Advance Engineering, ISSN:2349-2163, Issue 8, Vol 2, August 2018.

[6]https://mathworld.wolfram.com/TransitionMatrix.html

[7] https://en.wikipedia.org/wiki/Markov_chain