Pertanika J. Sci. & Technol. 21 (2): 541 - 554 (2013)

PERTANIKA JOURNALS

SCIENCE & TECHNOLOGY

Journal homepage: http://www.pertanika.upm.edu.my/

Application of Markov Chain in the PageRank Algorithm

Ravi Kumar, P.*, Alex Goh, K. L. and Ashutosh, K. S.

Department of Electrical and Computer Engineering, Curtin University, Sarawak Campus, 98009 Miri, Sarawak, Malaysia

ABSTRACT

Link analysis algorithms for Web search engines determine the importance and relevance of Web pages. Among the link analysis algorithms, PageRank is the state of the art ranking mechanism that is used in Google search engine today. The PageRank algorithm is modeled as the behavior of a randomized Web surfer; this model can be seen as Markov chain to predict the behavior of a system that travels from one state to another state considering only the current condition. However, this model has the dangling node or hanging node problem because these nodes cannot be presented in a Markov chain model. This paper focuses on the application of Markov chain on PageRank algorithm and discussed a few methods to handle the dangling node problem. The Experiment is done running on WEBSPAM-UK2007 to show the rank results of the dangling nodes.

Keywords: Markov chain, web graph, information retrieval, PageRank, transition probability, dangling page.

INTRODUCTION

Ranking algorithms or link analysis algorithms determine the success of the Web search engines as they calculate the importance and relevance of individual page on the World Wide Web. Examples of link analysis algorithms are HITS (Hyperlink Induced Topic Search), PageRank and SALSA (Stochastic Approach for Link Structure Analysis). These algorithms rely on the link structure of the Web pages. HITS (Kleinberg, 1999) developed by Jon Kleinberg, is a query depend algorithm, which calculate the authorities and hubs value of a page while SALSA (Lempel & Moran, 1999) algorithm combines the random walk feature in PageRank and the hub authority idea from HITS algorithm.

This paper, we focus on PageRank algorithm. PageRank (Brin & Page, 1998) is a query and content independent algorithm (Borodin *et al.,* 2005). Query independent means that the PageRank algorithm ranks all the pages offline after the crawler download

Article history:
Received: 26 December 2011
Accepted: 15 March 2012

E-mail addresses:
ravi2266@gmail.com (Ravi Kumar, P.),
alexgoh.kwangleng@gmail.com (Alex Goh, K. L.),
ashutosh.s@curtin.edu.my (Ashutosh, K. S.)
*Corresponding Author

ISSN: 0128-7680

and index the pages and the rank remains constant for all the pages. Content independent means the PageRank algorithm does not include the contents of a Web page for ranking rather it uses the link structure of the Web to calculate the rank. This PageRank algorithm is explained in section of Pagerank Algorithm. When a user types a query term on the search engine, the PageRank algorithm just finds the pages on the Web that matches the query term and presents those pages to the user in the order of their PageRank. It looks simple but the mathematical model behind is amazing. This paper explores the mathematical model behind the PageRank algorithm with experiments and results.

This paper is organized as follows. Next Section describes Markov chain and it's Mathematics. Section of Pagerank Algorithm explains PageRank algorithm using a sample Web graph. Section of Use of Markov Chain in Pagerank Algorithm describes how Markov chain is applied in the PageRank algorithm. Experiments and results are shown in Section of Experimental Results and Section of Conclusion concludes this paper.

MARKOV CHAIN

Introduction

Markov chain (Norris, 1996; Gao *et al*., 2009) is invented by A. A. Markov; a Russian Mathematician in the early 1900's to predict the behavior of a system that moves from one state to another state by considering only the current state. Markov chain uses only a matrix and a vector to model and predict it. Markov chains are used in places where there is a transition of states. It used in biology, economy, engineering, physics etc. But the recent application of Markov chain on the Google search engine is interesting and more challenging.

Markov Chain is a random process used by a system that at any given time $t = 1, 2, 3 \ldots n$ occupies one of a finite number of states (Gao et al., 2009). At each time t the system moves from state v to u with probability p_{uv} that does not depends on t. p_{uv} is called as *transition probability* which is an important feature of Markov chain and it decides the next state of the object by considering only the current state and not any previous states.

Transition Matrix

Transition Matrix T is an n x n matrix formed from the *transition probability* of the Markov process, where n represents the number of states. Each entry in the transition matrix t_{uv} is equal to the probability of moving from state v to state u in one time slot. So, $0 \leq t_{uv} \leq 1$ must be true for all $u,v = 1, 2, \ldots, n$. The following example shows a sample transition matrix of a 3 state Markov chain:

$$t_{uv} = \begin{bmatrix} 1/4 & 1/2 & 1/4 \\ 1/2 & 0 & 1/2 \\ 1/2 & 1/4 & 1/4 \end{bmatrix}$$

The *Transition matrix* must have the following properties:

- The matrix must be square and nonnegative matrix i.e. the number of rows and columns must be equal and the entries must be non negative. Each row and column represents a state.
- All the entries in the matrix represent probabilities, so, each entry must be between 0 and 1 inclusive.
- The sum of the entries in a row is the sum of the transition probabilities from a state to another state. So, the sum of the entries in any row must equal to one. This is called as *stochastic matrix.*

In the above *Transition matrix*, t_{uv}, we can easily see the probability of moving from one state to another state. For example $t_{3,2} = ¼$ i.e. the probability of moving from state 2 to state 3 is only 25%. This Markov chains are used to predict the probability of an event.

PAGERANK ALGORITHM

Web Graph

PageRank algorithm treats the Web as a directed labeled graph whose nodes are the pages and the edges are the hyperlinks between them (Broder *et al*., 2000). This directed graph structure in the Web is called as *Web Graph.* A graph *G* consists of two sets *V* and *E*. The set *V* is a finite, nonempty set of *vertices*. The set *E* is a set of pairs of vertices; these pairs are called *edges*. The notation *V*(*G*) and *E*(*G*) represent the sets of vertices and edges, respectively of graph *G*. It can also be expressed *G* = (*V*, *E*) to represent a graph. The graph in Fig.1 is a directed graph with 3 *vertices* and 3 *edges*.

The vertices *V* of *G, V*(*G*) = {1, 2, 3}. The Edges *E* of *G*, *E*(*G*) = {(1, 2), (2, 1), (2, 3), (1,3), (3,1)}. In a directed graph with *n* vertices, the maximum number of edges is *n*(*n*-1). With 3 vertices, the maximum number of edges can be 3(3-1) = 6.

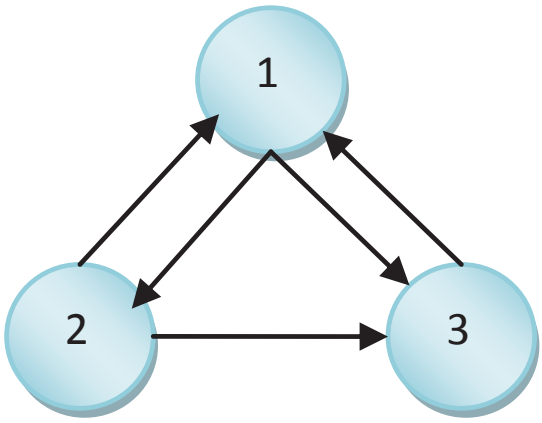

Fig.1: A Directed Web Graph *G*

There are a number of link based ranking algorithms (Brin & Page, 1998; Kleinberg, 1999; Lempel & Moran, 2000). Among them PageRank is the most popular link based ranking algorithm. PageRank algorithm and Google are developed by Brin and Page (1998) during their Ph D at Stanford University as a research project. The PageRank algorithm is the heart of the Google search engine. Google was introduced in the search engine business in 1998. Soon after its introduction, it became one of the most efficient search engine because it is a query independent and content independent search engine. It produces the results faster because it is query independent i.e. the Web pages are downloaded, indexed and ranked offline. When a user types a query on the search engine, the PageRank algorithm just finds the pages on the Web that matches the query term and presents those pages to the user in the order of their PageRank. PageRank algorithm uses only the link structure of the Web to determine the importance of a page rather than going into the contents of a page. PageRank provides a more efficient way to compute the importance of a Web page by counting the number of pages that are linking to it (in-coming links or backlinks). If an in-coming link comes from a reputed page, then that in-coming link is given a higher weighting than those in-coming links from a non-reputed pages. The PageRank *PR* of a page *p* can be computed by taking into account the set of pages *pa*(*p*) pointing to *p* as per the formula given by Page *et al.* (1999) is shown in equation [1] as follows:

$$PR(p) = d \sum_{q \in pa_p} \frac{PR_q}{O_q} + (1-d) \quad [1]$$

Here, d is a damping factor such that $0 < d < 1$ and O_q is the number of out-going links of page q.

Let us take an example of a simple Web graph with 3 nodes 1, 2 and 3 as shown in Fig. 1. The PageRank for pages 1, 2 and 3 can be calculated by using equation [1]. To start with, we assume the initial PageRank as 1.0 and do the calculation. The damping factor d is set to 0.85. PageRank calculation of page 1 is shown in equation [2] and [3].

$$PR(1) = (1-d) + d\left(\left(\frac{PR(2)}{O_2}\right) + \left(\frac{PR(3)}{O_3}\right)\right) \quad [2]$$

$$PR(1) = 0.15 + 0.85\left(\frac{1}{2} + \frac{1}{1}\right) = 1.425 \quad [3]$$

$$PR(2) = (1-d) + d\left(\frac{PR(1)}{O_1}\right) \quad [4]$$

$$PR(2) = 0.15 + 0.85\left(\frac{1.425}{2}\right) = 0.756 \quad [5]$$

$$PR(3)=(1-d)+d\left(\left(\frac{PR(1)}{O_1}\right)+\left(\frac{PR(3)}{O_3}\right)\right) \quad [6]$$

$$PR(3)=0.15+0.85\left(\left(\frac{1.425}{2}\right)+\left(\frac{0.756}{2}\right)\right)=1.077 \quad [7]$$

This PageRank computation continues until PageRank gets converged. This computation will be shown in the experiments and results section. Previous experiment (Page *et al*., 1999; Ridings & Shishigin, 2002) shows that the *PageRank* gets converged to a reasonable tolerance.

USE OF MARKOV CHAIN IN PAGERANK ALGORITHM

In the original PageRank algorithm by Brin et al., the Markov chain is not being mentioned. But the other researchers Langville and Meyer (2004b) and Bianchini *et al.* (2005) explored the relationship between PageRank algorithm and the Markov chain. This section explains the relationship between PageRank algorithm and Markov chain. Imagine a random surfer surfing the Web, going from one page to another page by randomly choosing an outgoing link from one page to go to the next one. This can some time lead to dead ends i.e. pages with no outgoing links, cycles around a group of interconnected pages. So, a certain fraction of the time, the surfer chooses a random page from the Web. This theoretical random walk is known as Markov chain or Markov process. The limiting probability that an infinitely dedicated random surfer visits any particular page is its PageRank.

The number of links to and from a page provides information about the importance of a page. The more back links or in-coming a page has, the more important the page is. Back links from more good pages carries more weight than back links from less important pages. Also if a good page points to several other pages then its weight is distributed equally to all those pages. According to Langville and Meyer (2004a), the basic PageRank starts with the following [8] to define the rank of a page p as $PR(p)$.

$$PR(p)=\sum_{q\in pa_p}\frac{PR_q}{|O_q|} \quad [8]$$

Where p is a Web page and $PR(p)$ is the PageRank of page p. pa is the set of pages pointing to p. O_q is the number of forward links of page q. The above [8] is a recursive equation. PageRank assigns an initial value of $PR_p^{(0)}=\frac{1}{n}$, where n is the total number of pages on the Web. The PageRank algorithm iterates as follows in [9].

$$PR_p^{(k+1)}=\sum_{q\in pa_p}\frac{PR_q^k}{|O_q|} \text{ for } k=0,1,2,..., \quad [9]$$

Where PR_q^k is the PageRank of page p at iteration k. The above equation [9] can be written in matrix notation. Let q^k be the PageRank vector at k^{th} iteration, and let T be the *transition matrix* for the Web; then according to Langville and Meyer (2004a),

$$q^{k+1} = Tq^k \qquad [10]$$

If there are n pages on the Web, T is an n x n matrix such that t_{pq} is the probability of moving from page q to page p in a time interval. Unfortunately, equation [10] has convergence problems i.e. it can cycle or the limit may be dependent on the starting vector. To fix this problem, Brin and Page build an irreducible* aperiodic+ Markov chain characterized by a primitive transition probability matrix.

The irreducibility guarantees the existence of a unique stationary distribution vector q, which becomes the PageRank vector. The power method with a primitive stochastic repetition matrix will always converge to q independent of the starting vector.

PageRank algorithm makes the hyperlink structure of the Web into a primitive stochastic matrix as follows. If there are n pages on the Web, let T be a n x n matrix whose element t_{pq} is the probability of moving from page p to page q in one step. The basic model takes $t_{pq} = 1/|O_q|$. If page q has a set of forward links, O_q, and normally all forward links are chosen equally as per the following equation [11].

$$t_{pq} = \begin{cases} \dfrac{1}{|O_q|} & \text{if there is a link from } q \text{ to } p, \text{ otherwise} \\ 0 \end{cases} \qquad [11]$$

The following Fig.2 shows a sample Web graph extracted from a University site. It shows a sample Web graph extracted from a University site contains 7 pages namely, Home, Admin, Staff, Student, Library, Dept and Alumni. We use this sample Web graph in our Markov analysis and PageRank calculation.

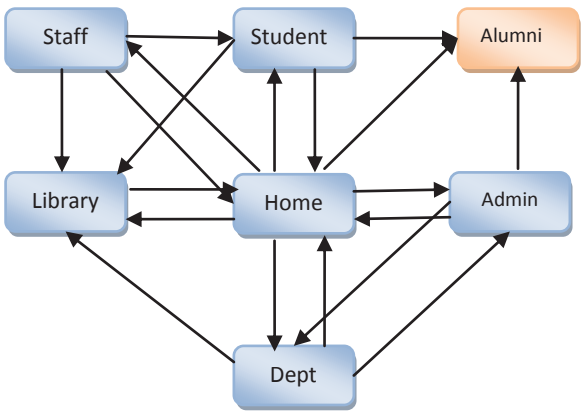

Fig.2: A sample Web Graph W of a University

*A Markov chain is irreducible if there is a non-zero probability of transitioning from any state to any other state.

+ An irreducible Markov chain with a primitive transition matrix is called as aperiodic chain.

Transition Matrix

The *transition matrix T* can be produced by applying in equation [11] to our sample Web graph on fig.2.

$$T = \begin{bmatrix} 0 & 1/3 & 0 & 1/3 & 1/3 & 0 & 0 \\ 0 & 0 & 1/3 & 1/3 & 1/3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1/6 & 1/6 & 1/6 & 1/6 & 0 & 1/6 & 1/6 \\ 0 & 0 & 1/3 & 0 & 1/3 & 0 & 1/3 \\ 0 & 0 & 0 & 1/3 & 1/3 & 1/3 & 0 \end{bmatrix}$$

In the *transition matrix T*, that row *q* has non-zero elements in positions that correspond to forward links to page *q* and column *p* has non-zero elements in positions that correspond to back links to page *p*. If page *q* has forward links, the sum of row is equal to 1.

In the *transition matrix*, if sum of any rows is zero that indicates that there is a page with no forward links. This type of page is called as *dangling node* or *hanging node.* Dangling nodes cannot present in the Web graph if it to be presented using a Markov model. There are a couple of methods to eliminate this dangling page problem. They are discussed using the *transition matrix* below:

Langville and Meyer (2004b) proposed a method to handle dangling pages is to replace all the rows with *e*/*n,* where *e* is a row vector of all ones and *n* is the order of matrix. In our example, the value of *n* is 7.

Using the above proposal, the sample Web graph in Fig.2 is modified as shown in Fig.3.

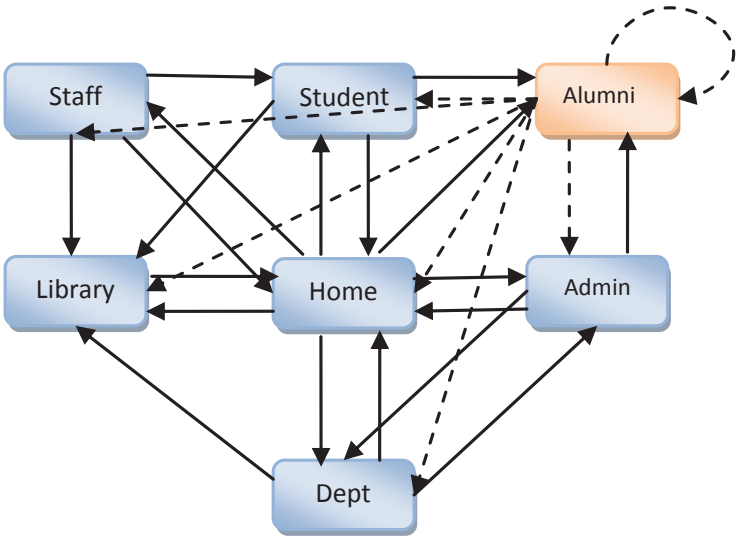

Fig.3: Modified Web Graph *W* using Langville and Meyer (2004b)

The new forward links from the Alumni page is shown using the dotted arrows. This makes the *transition matrix T* as stochastic as shown below:

$$\overline{T} = \begin{bmatrix} 0 & 1/3 & 0 & 1/3 & 1/3 & 0 & 0 \\ 0 & 0 & 1/3 & 1/3 & 1/3 & 0 & 0 \\ 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1/6 & 1/6 & 1/6 & 1/6 & 0 & 1/6 & 1/6 \\ 0 & 0 & 1/3 & 0 & 1/3 & 0 & 1/3 \\ 0 & 0 & 0 & 1/3 & 1/3 & 1/3 & 0 \end{bmatrix}$$

The row 3 in the *transition matrix* $\overline{T}$, (Alumni page) is connected to all the nodes and also connected back to it (shown in the dotted lines).

There is another proposal from Bianchini *et al.* (2005) and Singh *et al.* (2010) to connect a hypothetical node h_i with self loop and connect all the dangling nodes to the hypothetical node as shown in Fig.4. This method also makes the *transition matrix* as stochastic matrix.

In Fig.4, h_i is the hypothetical node with self loop (shown in blue dotted line) and the Alumni page is connected to it (shown in red dotted line) and the transition matrix for the modified graph is shown as follows:

$$\overline{T} = \begin{bmatrix} 0 & 1/3 & 0 & 1/3 & 1/3 & 0 & 0 & 0 \\ 0 & 0 & 1/3 & 1/3 & 1/3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1/6 & 1/6 & 1/6 & 1/6 & 0 & 1/6 & 1/6 & 0 \\ 0 & 0 & 1/3 & 0 & 1/3 & 0 & 1/3 & 0 \\ 0 & 0 & 0 & 1/3 & 1/3 & 1/3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

The last row and last column in the above transition matrix $\overline{T}$ is the hypothetical node h_i. The transition probability for the Alumni page in the modified graph in Fig.4 is 1. Now the Alumni page is no more a dangling page. Similarly for the hypothetical node h_i the transition probability is 1 because of the self loop. Now this Web graph on Fig.4 is also stochastic.

This stochastic property is not enough to guarantee that Markov model will converge and a steady state vector exists. There is another problem with is *transition matrix* $\overline{T}$ is that this matrix may not be regular. The general Web's nature makes the *transition matrix* $\overline{T}$ is not regular. In the graph, every node needs to be connected to every other node (irreducible). But in the real Web, every page is not connected to every other page i.e. it is not strongly connected. Brin et al. [1] forced all the entries in the transition matrix to satisfy $0 < t_{pq} < 1$ to make it regular. This ensures convergence of q^n to a unique, positive steady state vector.

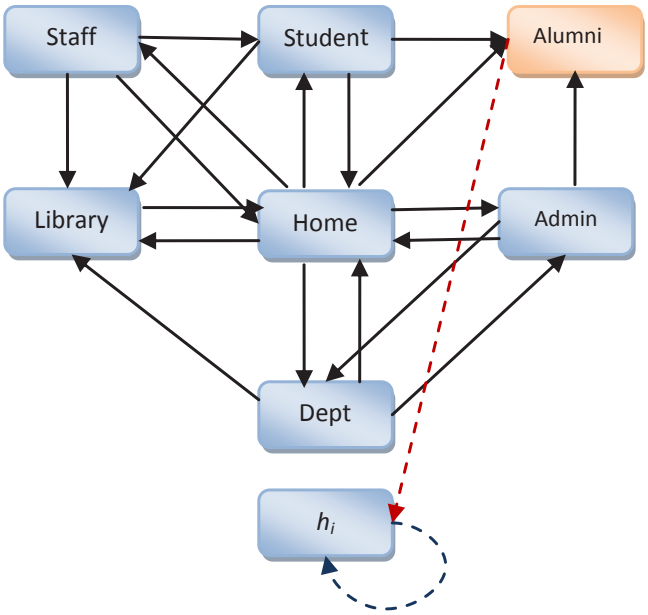

Fig.4: Modified Web Graph W using Singh *et at.* (2010)

Google Matrix

According to Langville and Meyer (2005), Brin and Page added a perturbation matrix $E = ee^t/n$ to make this stochastic irreducible matrix as Google matrix as shown in equation [12].

$$\overline{\overline{T}} = \alpha \overline{T} + (1-\alpha)E \qquad [12]$$

Where α is between 0 and 1. Google believed that this new matrix $\overline{\overline{T}}$ tends to better model the real-life surfer. In the real-life a surfer has 1-α probability of jumping to a random page on the Web i.e. by typing a URL on the command line of a browser and an α probability of clicking on a forward link on a current page. Many researchers (Brin & Page, 1998; Langville & Meyer, 2004b; Bianchini *et al*., 2005) say the value of α used by the PageRank algorithm of Google is 0.85.

We calculate the Google Matrix in equation [12] using the sample Web graph W by having a value of 0.85 for α and shown in the matrix $\overline{\overline{T}}$.

This matrix computation can be normalized to a stationary vector by calculating the powers of the transition matrix. At one stage of the calculation, the values of the matrix get stationary.

Those values are the PageRank scores for the 7 pages from the sample Web graph W. Assume after the 30th iteration the following are the stationary vector for our sample 7 page Web graph.

$$s = \begin{bmatrix} 0.043 & 0.334 & 0.047 & 0.341 & 0.441 & 0.041 & 0.041 \end{bmatrix}$$

$$\overline{\overline{T}} = 0.85\begin{bmatrix} 0 & 1/3 & 0 & 1/3 & 1/3 & 0 & 0 \\ 0 & 0 & 1/3 & 1/3 & 1/3 & 0 & 0 \\ 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1/6 & 1/6 & 1/6 & 1/6 & 0 & 1/6 & 1/6 \\ 0 & 0 & 1/3 & 0 & 1/3 & 0 & 1/3 \\ 0 & 0 & 0 & 1/3 & 1/3 & 1/3 & 0 \end{bmatrix} + 0.15\begin{bmatrix} 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 \\ 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 \\ 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 \\ 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 \\ 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 \\ 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 \\ 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 \end{bmatrix}$$

$$= \begin{bmatrix} 0.021 & 0.304 & 0.021 & 0.304 & 0.304 & 0.021 & 0.021 \\ 0.021 & 0.021 & 0.304 & 0.304 & 0.304 & 0.021 & 0.021 \\ 0.142 & 0.142 & 0.142 & 0.142 & 0.142 & 0.142 & 0.142 \\ 0.021 & 0.021 & 0.021 & 0.021 & 0.871 & 0.021 & 0.021 \\ 0.163 & 0.163 & 0.163 & 0.163 & 0.021 & 0.163 & 0.163 \\ 0.021 & 0.021 & 0.304 & 0.021 & 0.304 & 0.021 & 0.304 \\ 0.021 & 0.021 & 0.021 & 0.304 & 0.304 & 0.304 & 0.021 \end{bmatrix}$$

PageRank Interpretation of Stationary Vector

PageRank interprets the stationary vector in the following way. For example, a user enters a query in the Google search window requesting for word 1 and word 2. Then the search engine looks for the inverted index database with the word 1 and word 2. This database contains the list of all the words or terms and the list of documents that contains the words or terms (Langville & Meyer, 2005).

Assume the following documents lists are stored in the inverted index database for word 1 and word 2 as shown in Table 1.

So, the relevancy set for the user's query term word 1 and word 2 is {1, 2, 4, and 7}. The PageRank of these 4 documents are compared to find out the order of importance. According to our sample 7 page Web, 1 is the Staff page, 2 is the Student page, 4 is the Library page and 7 is the Dept. page. The respective PageRank scores are $s_1 = 0.043$, $s_2 = 0.334$, $s_4 = 0.341$ and $s_7 = 0.041$. This PageRank algorithm treats that document 4 (Library) page is most relevant to the given query term, followed by document 2 (Student), document 1 (Staff) and document 7 (Dept). When a user types a new query term, the inverted index database is accessed again and a new relevancy set is created. This is how the PageRank algorithm works in the Google search engine.

TABLE 1
Inverted Index Document List

Query Word/Term	Document List
Word 1	Document 2, Document 4 & Document 7
Word 2	Document 1 & Document 7

EXPERIMENTAL RESULTS

The dataset that is used in this experiment is WEBSPAM-UK2007, provided by Laboratory of Web Algorithmics, Università degli Studi di Milano with the support of the DELIS EU - FET research project (Yahoo! Research, n.d.). The collections contain 114549 hosts and among them, 49379 are dangling hosts. The distributions of hosts is as Fig.5.

The dataset is implemented with the algorithm in (Singh *et al.*, 2010) and shows the results of the dangling hosts. We actually show the rank results of the first dangling host to the last one. The rank results of the dangling hosts is described as in Fig.6.

With the hypothetical node, now the Web graph is stochastic. Fig.6 shows the rank results of the dangling hosts in ascending order. The results are calculated with the damping factor α of 0.85 and 50 iterations.

Label	No. of Hosts
Dangling Hosts	49379
Non-dangling Hosts	65150
Total	114529

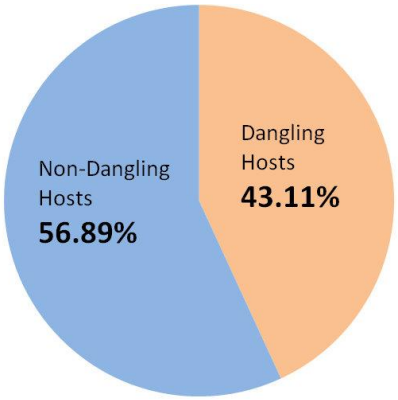

Fig.5: Distributions of Web Hosts

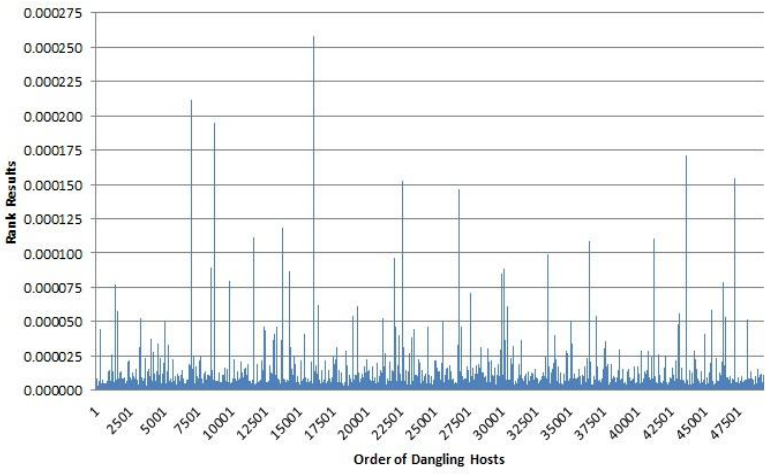

Fig.6: The rank results of the dangling hosts

CONCLUSION

This paper starts with the introduction of Markov chain and PageRank algorithm. Then the mathematics behind the PageRank algorithm is explained theoretically. This paper also brings anonymity about how the PageRank algorithm uses the Markov chain and transition matrix to calculate the relevancy set. This paper highlights the different adjustments done to make the Web graph into a Markov model. In that, the dangling node problem and the methods to handle the dangling nodes were also discussed and the mathematical solutions are given. A Markov model is created for a sample Web graph and the PageRank calculation is shown for the Markov model. We implemented the PageRank algorithm just to support our mathematical model and shown the results.

REFERENCES

Bianchini, M., Gori, M., & Scarselli, F. (2005). Inside PageRank. *ACM transactions on Internet Technology*, 2005.

Borodin, A., Roberts, G. O., Rosenthal, J. S., & TsaParas, P. (2005). *Link Analysis Ranking: Algorithms, Theory and Experiments*. In the Proceedings of the ACM Transactions on Internet Technology, Vol. 5, No 1, pp. 231-297.

Brin, S., & Page, L. (1998). The Anatomy of a Large Scale Hypertextual Web search engine. *Computer Network and ISDN Systems*, *30*(1-7), 107-117.

Broder, A., Kumar, R., Maghoul, F., Raghavan, P., Rajagopalan, S., Stata, R., Tomkins, A., & Wiener, J. (2000). Graph Structure in the Web. *Computer Networks : The International Journal of Computer and telecommunications Networking*, *33*(1-6), 309-320.

Gao, B., Liu, T. Y., Ma, Z., Wang, T., & Li, H. (2009). *A General Markov Framework for Page Importance Computation*. In the Proceeding of the 18th ACM conference on Information and knowledge management.

Kleinberg, J. (1999). Authoritative Sources in a Hyper-Linked Environment. *Journal of the ACM, 46*(5), 604-632.

Langville, A. N, & Meyer, C. D. (2004a). The Use of the Linear Algebra by Web Search Engines. *Bulletin of the International Linear Algebra*, *23*, December 2004.

Langville, A. N. & Meyer, C. D. (2004b). Deeper Inside PageRank. *Internet Mathematics*, *1*(3), 335-380.

Langville, A. N. & Meyer, C. D. (2005). *A Survey of Eigenvector Methods of Web Information Retrieval*. In the Proceedings of the SIAM, Vol. 47, No. 1, pp. 135—161, 2005.

Lempel, R., & Moran, S. (2000). *The stochastic approach for link–structure analysis (SALSA) and the TKC effect*. In Proceedings of the 9th World Wide Web Conference (WWW9). Elsevier Science. pp.387–401.

Norris, R. (1996). *Markov Chains.* Cambridge University Press. pp.1–4.

Page, L., Brin, S., Motwani, R., & Winograd, T. (1999). The Pagerank Citation Ranking: Bringing order to the Web. *Technical Report, Stanford Digital Libraries* SIDL-WP-1999-0120.

Ridings, C., & Shishigin, M. (2002). PageRank Uncovered. *Technical Report*, 2002.

Singh, A. K., Kumar, P. R., & Alex, G. K. L. (2010). *Efficient Algorithm to handle Dangling Pages using Hypothetical node.* In the Proceeding of 6th IDC2010, Seoul, Korea, 2010.

Yahoo! Research. (n.d.). *Web Spam Collections*. Retrieved on July 2011 from http://barcelona.research.yahoo.net/webspam/datasets/ Crawled by the laboratory of Web Algorithmics, University of Milan, http://law.dsi.unimi.it/.