Web page recommendation using a stochastic process model

B. J. Park[1], W. Choi[1] & S. H. Noh[2]
[1]*Computer Science Department, Kwangwoon University, Korea*
[2]*Samsung Electronics, Korea*

Abstract

Recommending Web pages based on the access patterns by previous visitors to a user visiting some Web site can be helpful if they are is not familiar with the site. To provide a proper recommendation service for users, it is necessary to efficiently mine the Web page access patterns from a huge amount of Web server log data. This paper presents an efficient Web page recommendation scheme using a stochastic process model based on the time-homogeneous DTMC (Discrete-Time Markov Chain). It views a series of Web page access as a discrete-time stochastic process and constructs a state transition matrix based on the relative frequency of moving from one page to another in the Web site. Then it uses the initial state transition matrix to compute the probability for each Web page to be accessed after some number of page references from the starting Web page and selects a page with the largest probability value for recommendation. Although this process involves a simple matrix multiplication, its computational overhead could be expensive if the number of pages in the Web site is very large. Our approach employs the PageGather algorithm to find clusters of closely related ones among all the Web pages and uses each of these clusters as a state in the state transition matrix. As a result, the size of the transition matrix can be considerably reduced and so is the computation time without sacrificing the effectiveness of recommendation too much. We demonstrate the effectiveness and efficiency of our recommendation scheme with a series of experiments.
Keywords: Web mining, Web page recommendation, stochastic process model.

WIT Transactions on Information and Communication Technologies, Vol 37, © 2006 WIT Press
www.witpress.com, ISSN 1743-3517 (on-line)
doi:10.2495/DATA060241

1 Introduction

Recommending Web pages based on the access patterns by previous visitors to a user visiting some Web site can be helpful if they are not familiar with the site [1]. Web users entering a certain Web site usually visit a series of pages in the site before they leave it after having achieved their goals for the visit. To provide a proper recommendation service for users, it is necessary to efficiently mine the Web page access patterns from a huge amount of Web server log data.

Mining the Web server log containing data such as the IP address, the access time and date, and the requested URL in each entry can discover useful time-dependent behaviours and access patterns of the Web users. For this type of Web usage mining, there has been extensive research work based on association rules and sequential patterns.

This paper proposes an approach to Web page recommendation using a stochastic process model based on the time-homogeneous discrete-time Markov chain (DTMC) which recommends to the users a set of Web pages that are expected to be referenced after some number of page access from the current page.

The rest of this paper is organized as follows: In section 2, we discuss some of the existing approaches to the Web usage mining and the stochastic process model based on DTMC. In section 3, we present our approach to Web page recommendation using the stochastic process model. In section 4, we show some experimental results that demonstrate the effectiveness of our approach. Section 5 draws some conclusions based on the experimental results.

2 Background and related work

Most of existing approaches to Web page recommendation based on Web usage mining use the pattern discovery techniques such as the association rule discovery, the sequential pattern discovery, and the path analysis [2, 3, 7, 8, 10, 11]. Our approach, however, uses a stochastic process model based on DTMC. This section discusses some of the existing approaches and the modelling of Web access behaviours as a DTMC.

2.1 Access pattern discovery techniques

Association rules for Web pages describe associations between pages. For instance, “If a user visits page A, then they visit page B as well” is a typical association rule that can be found by an association rule mining algorithm for the Web. There has been extensive work on the association rule discovery, including Apriori, DHP, Sampling, FUP, and DIC [7]. Since there is no ordering relation between the Web pages in the association rules, they can be useful for recommending Web pages not sequentially related with the current page. For instance, they are appropriate for recommending a set of pages for the book titles to the online shoppers currently viewing a page on some book in a certain category.

Sequential patterns are similar to association rules, but there is an ordering between the items in the pattern. For instance, "Users visit page B after they visit page A" is a typical sequential pattern for the Web page access. Various approaches to sequential pattern discovery have been proposed including the discovery of frequent episodes from an event sequence [5], FAP [7], FP [10], etc.

Since the volume of Web log data collected for even a moderate-sized Web site during some period of time could be huge, it will be very time-consuming to mine all the data over and over again as the new log data are collected. Thus, some form of incremental mining would be especially helpful for Web usage mining. Since the incremental mining only mines newly added data and updates the overall pattern by combining the newly found ones, it can greatly reduce the overall time required to find the up-to-date patterns if the existing patterns and newly found ones could be easily combined.

2.2 A stochastic process model based on discrete-time Markov chain

The Web users visiting some Web site usually move from one page to another by following the hyperlinks connecting pages. We can model these access behaviours as a stochastic process based on time-homogeneous DTMC. Given m Web pages, $w_1, w_2, \ldots, w_m$, ($m \geq 0$) in a Web site, we can associate a probability p_{ij} with each access sequence from w_i to w_j , where $p_{i1} + p_{i2} + p_{i3} + \ldots + p_{im} = 1$ for each i ($1 \leq i \leq n$). In this case, probability p_{ij} is defined as a relative frequency of the access from w_i to w_j over the total number of accesses to $w_1, w_2, \ldots, w_m$ from w_i. Then, we can construct a state transition matrix P for time-homogeneous discrete time Markov chain as follows:

$$P = \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1m} \\ p_{21} & p_{22} & \cdots & p_{2m} \\ \cdots & \cdots & \cdots & \cdots \\ p_{m1} & p_{m2} & \cdots & p_{mm} \end{bmatrix} \quad (1)$$

With this state transition matrix P and the initial distribution of DTMC $a^{(0)} = a$, the n^{th} state could be computed as follows:

$$a^{(n)} = a^{(0)} P^{(n)} = a P^n \quad (2)$$

3 Web page recommendation using a stochastic process model

In this section, we present a Web page recommendation scheme using a stochastic process model based on DTMC. Unlike most of other recommendation systems that recommend a set of pages which are likely to be accessed as the next page, our recommendation technique can be used to recommend a set of pages which are likely to be accessed as the last page after

some number of page references. Recommending a set of Web pages that are going be eventually referenced for the users can directly lead the users to the pages that are likely to be their final destinations. For instance, for an internet shopping mall site, most users will leave the site after they complete the transaction for the product they wished to purchase. Or if some object becomes popular for download in some Web site, it is likely that the page containing the object will be the last page to be accessed for the most users. In the following subsections, we present the proposed Web page recommendation process in detail.

3.1 Web page recommendation process

The overall process for Web page recommendation based on a stochastic model is described in Figure 1.

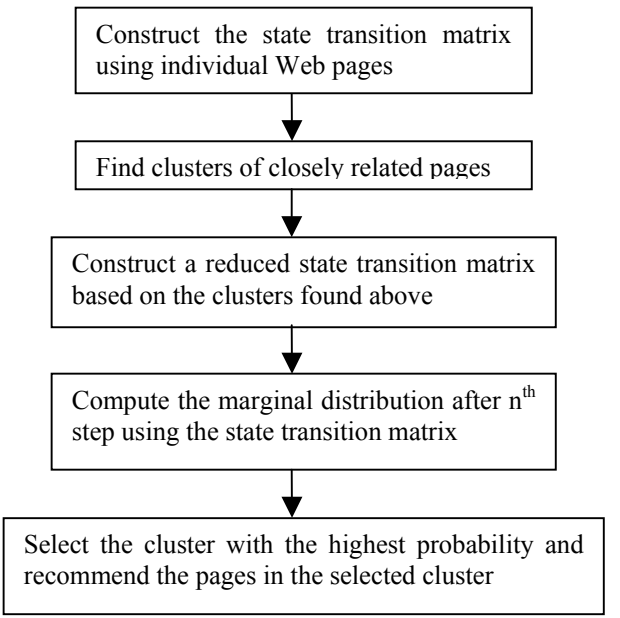
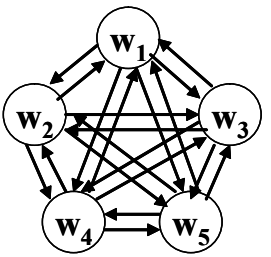


Figure 1: Web page recommendation process using a stochastic process model.

3.1.1 Construction of a state transition matrix using individual Web pages

After some preprocessing of the Web server log, one can identify distinct user sessions and count the number of references from one page to another in all identified sessions. These counts could be converted to probability values, each of which constitutes a cell value in the state transition matrix. For instance, given a Web site with 5 Web pages, the state transition matrix can be constructed as in Figure 5.



	W_1	W_2	W_3	W_4	W_5
W_1	0	0.1	0.4	0.3	0.2
W_2	0.5	0	0.1	0.1	0.2
W_3	0.1	0.2	0	0.3	0.4
W_4	0.7	0.1	0.1	0	0.1
W_5	0	0	0.5	0.5	0

Figure 2: Construction of state transition matrix.

3.1.2 Clustering of closely related Web pages

If a Web site consists of a large number of Web pages, the size of the constructed state transition matrix can be huge, which makes the entire computation process take a long time to get the final result. Thus, to reduce the size of the state transition matrix without loosing the effectiveness of recommendation too much, we employ a clustering algorithm called PageGather [1] to find the clusters of closely related Web pages. The PageGather algorithm uses the state transition matrix constructed in the previous step to generate a graph structure and finds the cliques(clusters) from it. The following figure shows an example of the clusters found for the Web pages in Figure 2.

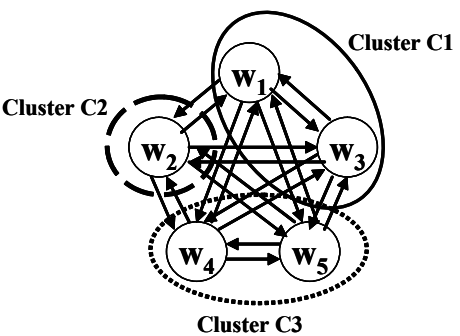


Figure 3: Clustering of closely related Web pages.

The PageGather algorithm performs well for a large set of Web pages compared with the K-means algorithm or the HAC algorithm [1].

3.1.3 Reduction of the state transition matrix

Once a set of clusters has been found in the previous step, each cluster, not the individual pages, now could be a state in the state transition matrix. Since the number of clusters is considerably smaller than that of the individual pages, the size of the new state transition matrix could be reduced considerably. For instance, for the previous Web site example with 5 pages, the size of the original

state transition matrix is 5 ×5, while the size of the cluster-based state transition matrix is 3 ×3. The following figure shows an example of the reduced state transition matrix constructed after clustering.

Table 1: A reduced state transition matrix based on clusters

	C1	C2	C3
C1	0.4	0.4	0.2
C2	0.5	0.3	0.2
C3	0.1	0.5	0.4

Each cell value of the reduced state transition matrix has to be computed from the original matrix with the additional information about the number of outgoing counts on the links for each page.

3.1.4 Selection of a cluster of pages to recommend

Now that we have a reduced state transition matrix based on the clusters of closely related pages, the next step is to select the most probable cluster to be referenced finally. This step involves the computation of $a^{(n)}$ where a is the initial state vector representing what cluster of pages the user is currently referencing. To compute $a^{(n)}$, it uses the formula (2) where a is k-dimensional vector (k is the number of clusters, that is, the number of the rows in the state transition matrix) and P is the reduced state transition matrix. For instance, in case of our example Web site, if the current page belongs to clusters 3, the initial state vector a is (0, 0, 1) and the following is the state after the n steps from the current page.

$$(0,0,1)\times\begin{bmatrix}0.4 & 0.4 & 0.2\\ 0.5 & 0.3 & 0.2\\ 0.1 & 0.5 & 0.4\end{bmatrix}^{n} \tag{3}$$

After the computation is done, the cluster selected for recommendation is the one with the highest probability value among the elements in the resulting vector.

4 Experimental results

We have conducted a series of experiments to evaluate the performance of our recommendation scheme. We used a Web server log from a price comparison site named www.bb.co.kr for the experiments. This site receives about 28,000 page requests per hour in average and consists of more than 2500 Web pages. We used the Web log collected for 2 days for our experiments. The log data for the first day are used as a training data and the second day's as the test data. After some preprocessing of the Web log file, 2032 Web pages were found to be referenced by users. Thus, the size of the original state transition matrix constructed from the log analysis is 2032 × 2032. And the total of 639 clusters

were formed after clustering, so the size of the reduced state transition matrix becomes 639× 639.

In order to measure the quality of our recommendation scheme, we defined the concept of "effectiveness" for each recommendation. Given some initial page *w* and a positive integer *n*, the *effectiveness* of recommendation R={r_1,r_2, …, r_k} with respect to *w* and *n* (where r_1,r_2, …, and r_k are individual pages) is defined as the ratio of the number of sessions whose initial page is *w* and final page belongs to R to the total number of sessions whose initial page is *w*.

A series of experiments have been conducted to answer the following questions regarding the performance of our scheme:

[1] It is worthwhile to recommend a set of pages that are likely to be accessed as the final page after a certain number of page visits instead of recommending pages that are popular for the immediate visit?

[2] What are the effects of reducing the number of states (by clustering) in the state transition matrix on the overall effectiveness of our recommendation scheme?

[3] How does the performance of our scheme using a single matrix for the entire period of time available for training compare to the one using multiple matrices for different time periods?

In the next subsections, we describe the experimental results with regard to the above questions.

4.1 Experiment 1: 1-step vs. n-step ahead recommendation

For the experiments, we arbitrarily selected 200 Web pages out of those 2032 pages to use as the current start page since it took too much time to experiment with all of 2032 pages. We measured the overall effectiveness of our recommendation scheme by taking the average of those 200 cases for each of 1-step, 5-steps, and 10-steps ahead recommendation. As shown in Table 2, recommending pages 5-steps ahead resulted in the best effectiveness among 3 trials with a different number of steps. It is notable that an unnecessarily large number of steps ahead cause the average effectiveness to get worse.

Table 2: Average effectiveness for each case with different number of steps

# of steps ahead	1	5	10
Average Effectiveness	33.8%	40.5%	21.2%

4.2 Experiment 2: effects of the matrix size reduction by clustering

Since we employed a clustering approach to reduce the size of the state transition matrix and hence to shorten the computation time for matrix multiplication, we

have conducted an experiment to see its effects on the performance. For the experiment, we chose as candidates for the initial page a total of 40 pages out of the 200 pages mentioned above: 20 with the highest effectiveness values and the other 20 with the lowest effectiveness values. We have experimented for a total of 40 runs of execution to get the average run time and effectiveness with and without clustering. For this experiment, we set the number of steps ahead to 5 and used a PC with a Pentium 4- 2GHz CPU and 512MB of RAM. Table 3 shows the performance of our scheme.

Table 3: Performance comparison of the original matrix and the reduced matrix.

	Original matrix	Reduced matrix
Average runtime	15 min 32 sec	6 sec
Average Effectiveness	46.5 %	41.5 %

As we can see in Table 3, the reduction of about 10% in the average effectiveness has been observed while the average runtime has been greatly reduced using a reduced matrix.

4.3 Experiment 3: single matrix vs. multiple matrices

Since the Web access patterns of different groups of Web users for different time periods of a day could be different, we have conducted two types of experiments with the Web log: one with a single state transition matrix for the time period of all day and the other with a distinct state transition matrix for each time period a day reflecting different access patterns for different time periods. For convenience, we selected 3 different time periods of a day, namely 13:00- 16:00, 17:00 – 20:00, and 21:00 – 24:00 and constructed a state transition matrix for each time period based on the log data for the first day. For this experiment, we set the number of steps ahead for recommendation to 5 and used the 3 corresponding portions of the entire log as test cases to measure the performance of the individual matrices. As shown in Table 4, the average effectiveness for all of the three cases has been improved by 11-15% relatively (from 40.5 to 45.4 – 47.3).

Table 4: Single matrix vs. multiple matrices.

Time period	13:00 - 16:00	17:00 - 20:00	21:00 - 24:00
Average Effectiveness	47.1%	45.4%	47.3%

5 Concluding remarks and future work

In this paper, we presented a novel Web page recommendation scheme using a stochastic process model based on DTMC. Unlike most of other

recommendation systems that recommend a set of pages which are likely to be accessed as the next page, our recommendation scheme can be used to recommend a set of pages which are likely to be accessed as the last page after some number of page references. Recommending a set of Web pages that are going be eventually referenced for the users can lead the users directly to the pages that are likely to be their final destinations.

With a series of experiments, we demonstrated that our scheme could recommend a set of destination pages for the users both effectively and efficiently. We also showed that the effectiveness of our scheme could even be improved by exploiting a more sophisticated approach based on a different transition matrix for each different time period.

As for future work, we are currently investigating on the cases where our scheme didn't work well because the clustering process did not yield a substantial reduction in the size of the state transition matrix. These cases arise when the link structure of the Web pages is simplistic or when the Web site has a hierarchical structure of Web pages with a large number of terminal pages, causing the size of each cluster to become smaller. Work on developing an efficient clustering technique which is applicable to various types Web structure is needed.

References

[1] Perkowitz, M., & Etzioni, O., Adaptive Web Sites: Automatically Synthesizing Web Pages, *Proc. of the 15th national/10th conference on Artificial intelligence/Innovative applications of artificial intelligence*, pp.727-732, 1998.

[2] Agrawal, R. & Srikant, R., Fast Algorithms for Mining Association Rules, *Proc. of the 20th VLDB Conference*, Santiago, Chile, pp. 487-499, 1994.

[3] Srivastava, J., Cooley, R., Deshpande, M. & Tan, P-T., Web Usage Mining: Discovery and Applications of Usage Patterns from Web Data, *SIGKDD Explorations*, 1(2), pp.12-23, 2000.

[4] Park, J. S., Chen M.-S., & Yu, P., An Effective Hash-based Algorithm for Mining Association Rules, *Proc. of ACM SIGMOD Conference on Management of Data*, pp.175-186, 1995.

[5] Toivonen, H., Sampling Large Database for Association Rules, *Proc. of the 22nd VLDB Conference*, pp.134-145, 1996.

[6] Cheung, D.W., Han, J., Ng, V. & Wong, C., Maintenance of Discovered Association Rules in Large Databases: An Incremental Updating Technique, *Proc. of Int'l Conference on Data Engineering*, pp.106-114, 1996.

[7] Wang, X., Ouyang, Y., Hu, X. & Zhang, Y., Discovery of User Frequent Access Patterns on Web Usage Mining, *Proc. of the 8th International Conference on CSCW in Design*, pp. 755–769, 2003.

[8] Chen, M. S., Park, J. S. & Yu, P. S. , Data Mining for Path Traversal Patterns in a Web Environment, *Proc. of the 16th International Conference on Distributed Computing Systems,* pp.385-392, 1996.

[9] Kulkarni, V. G., *Modeling and Analysis of Stochastic Systems*, Chapman & Hall: London, UK, 1995

[10] Han, J., Pei, J., & Yan, X, Sequential Pattern Mining by Pattern Growth: Principles and Extensions, *Recent Advances in Data Mining and Granular Computing, (Mathematical Aspects of Knowledge Discovery)*, W. Chu and T. Lin (eds.), Springer Verlag: Heidelberg, pp.183-220, 2005.

[11] Han, J., Pei, J., Yin, Y., & Mao, R., Mining Frequent Patterns without Candidate Generation: A Frequent Pattern Tree Approach, *Data Mining and Knowledge Discovery: An International Journal*, 8, Kluwer Academic Publishers, pp.53-87, 2004.