# Adaptive Web Recommendation for New Navigation Trends

Yi-Jen Ian Su  Hewijin Christine Jiau  Shang-Rong Tsai

Department of Electrical Engineering
National Cheng Kung University
Tainan, Taiwan 701

Contact:

Yi-Jen Ian Su
Department of Electrical Engineering
National Cheng Kung University
Tainan, Taiwan 701
iansu@eembox.ncku.edu.tw
Phone: 886-6-2757575 ext. 62400-717
Fax: 886-6-2345482

## Abstract

All website managers want to attract more users. Web Usage Mining is a pattern-identifying device widely adopted for this purpose, but it has proved to be inadequate as the demand for on-line work increases. Though Web Usage Mining helps identify the users' surfing habits and does it effectively, the process of pattern discovery is time-consuming as it requires processing all access records occurred in the past days or hours. New trends of web browsing, however, are forming all the time. Web content needs to be updated accordingly. If the new content is hot or noticeable, users' behavior will change instantly. E-commerce or news websites are concrete examples. On-line users are practical and goal-oriented. They will not use a recommendation engine again if the information it provides no longer meets their needs. It is therefore essential that a recommendation engine have a high level of accuracy.

In this paper we propose a recommendation engine that can quickly respond to new navigation trends and provide users with the best suggestions on hyperlinks. This engine is especially effective when there is a change in web content but this information has not been assimilated into regular patterns yet. Ultimately, the research aims to equip websites with facilities that can customize users' needs automatically and efficiently.

**Keywords**:Web Usage Mining, Recommendation Engine

**Workshop**:Workshop on Databases and Software Engineering

## 1. Introduction

Just as data mining [1] helps us discover regular patterns hidden in a huge amount of data, web usage mining assists us in analyzing the users' browsing behavior when there are a large number of user access log records. Consider, for example, the case of YAHOO. This famous website has 16.6 million users every day and generates as much as 48GB access log records each hour [2]. Faced with such a big flow of clickstream, we could very well give the user a good recommendation of what to browse next simply based on his or her profile. It is, however, very hard to identify each user precisely, since most of them adopt anonymity when they browse websites. How to give timely and personalized linking recommendations while handling a huge amount of data at the same time hence defines one of the biggest challenges for website servers.

Generally, the recommendation system can be divided into two phases based on operation time: the off-line phase and the on-line phase, as illustrated in Fig. 1. First, in the off-line phase, web usage mining completes preprocessing and pattern discovery. There are two major data mining operations that deal with web access log file and website topology respectively. Second, based on each user's clickstream and the patterns discovered in the off-line phase, the recommendation engine gives the on-line user the best next-step recommendation in the on-line phase.

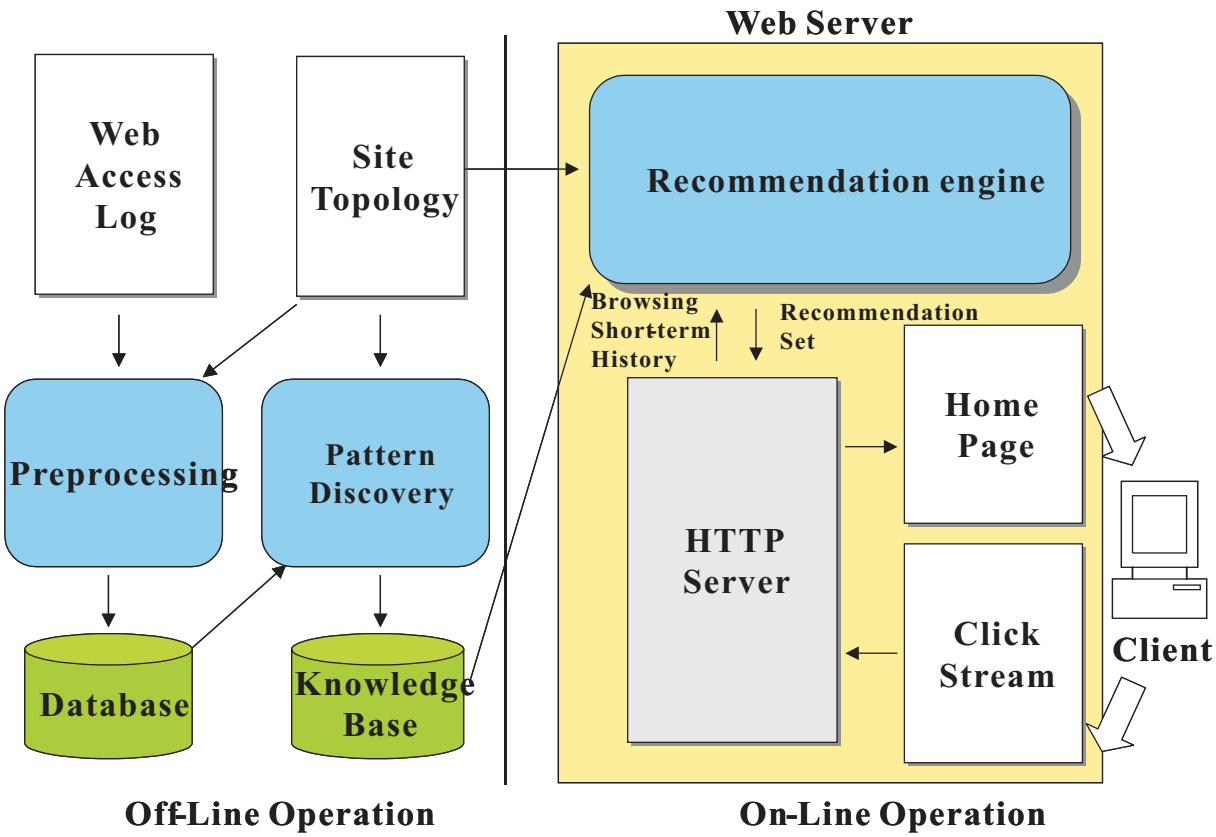


**Figure 1. Operation Sequence of Web Usage Mining**

The preprocessing job can be divided into four steps in the operation sequence: data cleaning, session identification, traveling path completion, and transaction identification [3, 4]. The major goal of pattern discovery is to look for regular navigation patterns in the database of preprocessed access log records.

The recommendation engine takes both regular patterns and user's clickstream as input and generates a candidate set for a user by matching the user's short-term browsing behavior with the discovered patterns. Because the recommendation engine is an on-line process, its accuracy, efficiency, and speed must all be taken into consideration.

In this paper we propose a dynamically adaptive recommendation engine that is especially useful for those web servers who update their site content frequently, such as e-commerce or news websites. Our recommendation engine identifies and utilizes the trends of on-line users' browsing behavior to generate appropriate recommendation sets in near real time.

The rest of this paper is organized as follows. In Section 2 we describe several web mining related techniques that are referenced in our research. In Section 3 we give a detailed explanation of the new recommendation engine. In Section 4 we implement the whole system and examine its performance. Finally, Section 5 summarizes and concludes our study.

## 2. Related Work

### 2.1. Pattern Discovery

In this research we apply two data mining techniques to web usage mining: association rules [1] and clustering [1]. Association rules are used to represent the relationship between two web pages as hyperlinks. The clustering divides users' session into clusters in order to speed up recommendation. FM models [5] are applied to assess the features of the session, and a similarity function is used to measure whether the session belongs to an existing cluster or is forming a new cluster.

#### 2.1.1 Association Rules

Basically association rules are used to find the relationship between data attributes. The Market Basket Analysis exemplifies this method well. Suppose the set $\Omega$ represents all of the items that appear in an event, $\Omega$ = { $A_1$, $A_2$, ..., $A_k$ }, we want to find all the combinations between $A_i$ and $A_j$ that can satisfy

$$Supp.(A_i, A_j) = P(A_i \bigcap A_j) = \frac{Count(A_i \bigcap A_j)}{Count(A_{all})} (1)$$

$$Conf.(A_i, A_j) = P(A_j \mid A_i) = \frac{Count(A_i \bigcap A_j)}{Count(A_i)} (2)$$

Here $A_i$ and $A_j$ stand for the items that customers would buy. Both (1) and (2) can be accepted because they satisfy both the support threshold and the confidence threshold. Support refers to the appearance probability in all transactions, whereas confidence refers to the probability of having $A_i$ appearing ahead of $A_j$.

Apriori [6] is the most famous algorithm. By setting up threshold values, Apriori removes a considerable number of items from the candidate itemset, thus effectively reducing the problem domain for association rules. A hash function is adopted in [7] to speed up the searching process of association rules. As a result, [7] generally has much better performance than Apriori.

#### 2.1.2 Clustering

Clustering groups the data into clusters which include objects with high similarity. In this research the clustering method compares all of the transaction to search for similar features. To which cluster each transaction belongs will not be known until a similarity measurement is taken. All the transaction belonging to the same cluster manifest similar browsing behavior. Because we can not identify exactly who each user is, the recommendation engine uses similar browsing patterns and the user's clickstream to predict the browsing targets and gives recommendations accordingly. In anonymous web usage mining, we must, first of all, model each anonymous user's behavior from web access log file and then construct clusters based on the results of similarity measurement. When a user does on-line browsing, the recommendation engine uses the current user's short-term browsing behavior to justify which cluster this behavior pattern belongs to, then uses this cluster's browsing sequence to give recommendations. At present most studies of collaborative filtering algorithm are successful on the Markov model (probabilistic based) [8] and the Vector model (distance based) [9].

### 2.2. Similarity Measurement

In general, similarity measurement is adopted to help decide which cluster current data belong to. There are two popular measuring methods for calculate similarity: one uses the cosine measure and the other uses the Euclidean distance.

#### 2.2.1 Vector Angle

The vector angle uses an included angle of two vectors to justify the similarity of these two vectors. If the included angle is large, then the similarity is low; on the other hand, if the

included size is small, then the similarity is high. The formula of the vector angle is

$$Angle(\vec{X},\vec{Y}) = \frac{\vec{X}\cdot\vec{Y}}{|\vec{X}||\vec{Y}|} = \frac{\sum_{i=1}^{N} X_i Y_i}{\sqrt{\sum_{i=1}^{N} X_i^2}\sqrt{\sum_{i=1}^{N} Y_i^2}} (3)$$

#### 2.2.2 Euclidean Distance

The Euclidean distance uses the difference between two vectors to justify the similarity. Formula (4) is Euclidean Distance, which can calculate the difference between two vectors.

$$Distance(\vec{X},\vec{Y}) = |\vec{X}-\vec{Y}| = \sqrt{\sum_{i=1}^{N}(X_i - Y_i)^2} (4)$$

#### 2.2.3 Projected Pure Euclidean Distance (PPED)

PPED [5] substantially improves the time complexity of Euclidean Distance's difference calculation (formula (4)). When a user's transaction (or session) is represented at Feature Matrices, the transaction is very likely to form a spare matrix. Therefore if we use Euclidean Distance to do similarity measurement, it would cause many useless calculations.

The recommendation engine weighs what to recommend the on-line user based on the user's short-term behavior and stored regular patterns in the knowledge base. PPED therefore uses the user's short-term browsing transaction to justify which cluster the transaction belongs to and then compares the transaction against the cluster's common behavior pattern to generate a recommendation that meets the requirement of real-time operation. In order to use the browsing behavior of shorter time intervals, like sliding window, to generate recommendations for the current user, PPED nice partial order matching method to speed up the process of similarity measurement.

$$PPED(\vec{X},\vec{Y}) = \sqrt{\sum_{i=1, X_i\neq 0}^{N}(X_i - Y_i)^2} (5)$$

### 2.3. Feature Matrices Model

The Feature Matrices (FM) Model [5] is designed to solve the problem of anonymous web usage mining. Basically the FM model is an extension of the vector model. The FM model takes problem domain into account in order to do effective quantification. Three features are

already taken into account in web usage mining: hit count (H), browsing sequence (S), and visit time (T). Hit count and browsing sequence are spatial features, whereas visit time is a temporal feature. First, we partition each session into transactions of different web categories in order to reduce the problem domain. Then each transaction is mapped into the FM model with the help of the three features mentioned above.

The entire set of feature matrices generated for a transaction constitutes its FM model:

$$U^{fm} = \{M_{r^2}^H, M_{r^2}^S, M_r^T\}(6)$$

$M_{r^n}^F$ means n-dimension feature matrix with r rows, which records the F feature values for all order-n segments of Universal Feature Matrices.

#### 2.3.1 Cluster Mode

When the similarity measurement is completed, it will be decided which cluster the current transaction belongs to. By adding a transaction into its cluster, we adopt the method of matrix content average with formulas (7), (8).

$$M^F = \frac{1}{N}\sum_{i=1}^{N} M_i^F(7)$$

$$M_{new}^F = \frac{1}{N+1}(M_{old}^F + M_{i+1}^F)(8)$$

## 3. System Architecture

A few ideas led to the design of a dynamically adaptive recommendation engine [10]. First, the content of the website might be modified all the time. Web pages could be added, removed, or updated by the site organizer. If regular patterns were discovered yesterday and a new web page is added to the website this morning, then it is impossible to generate a prediction for the new web page from these found patterns. Second, the users of different time intervals have different browsing habits. For example, a lot of the morning users may be housewives that want

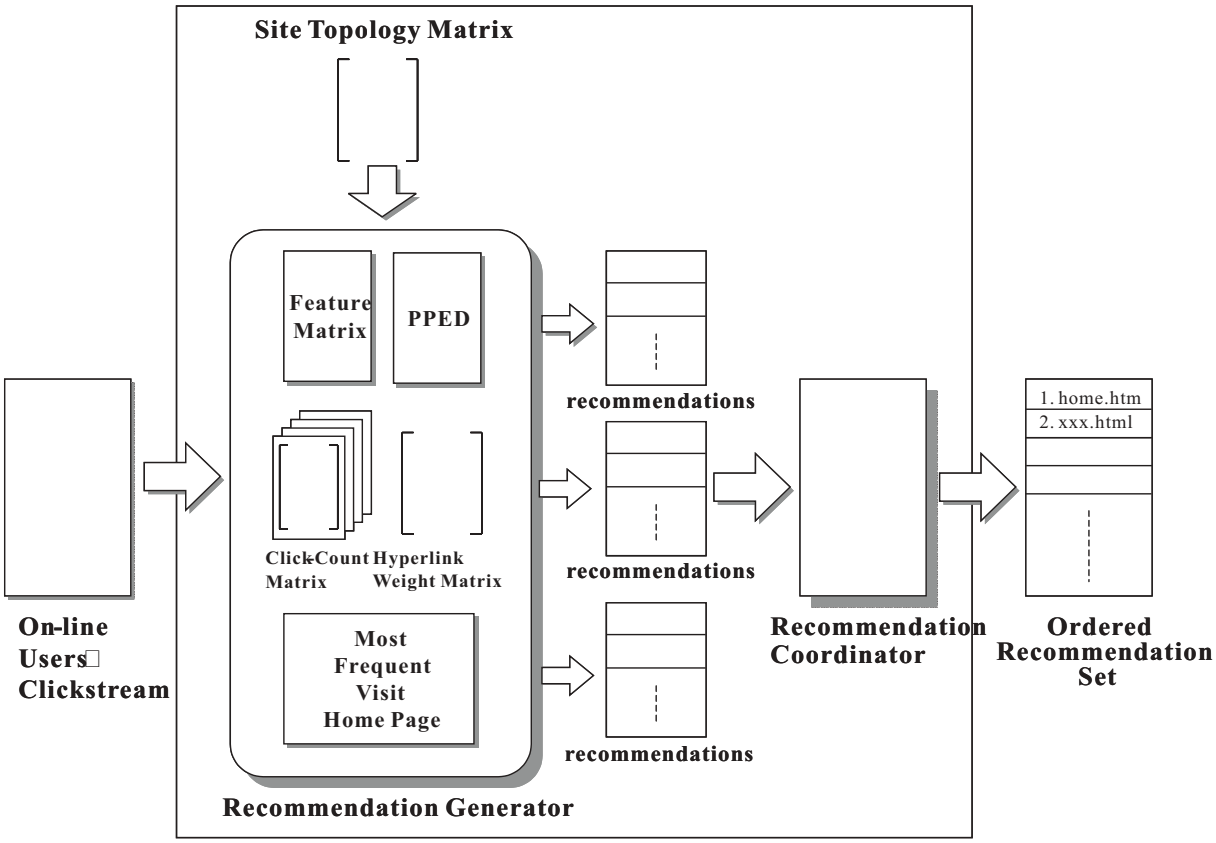


**Figure 2. The Architecture of The Recommendation Engine**

to find some information about e-commerce; however, a considerable number of the afternoon web browsers may be students that want to look up information for their school work. All the users have different purposes in mind when they browse the web at a different time during the day. Finally, the trend of long-term user behavior is different from that of short-term user behavior. For example, when a hot news web page is added to a news web server, short-term user behavior would very likely manifest a new trend to click on the hyperlink of this web page, but it is hard to generate a prediction from long-term user behavior. That is why, to enhance the accuracy of the recommendation engine, we use the patterns of both short-term and long-term user behavior to give on-line user recommendations.

If the recommendation engine only discovers regular patterns of user behavior from the web access log off-line and uses them to predict user's next step and generate recommendations on-line, it would not be able to meet all of the needs of the current user. Once a user thinks that

the recommendation set can not satisfy him, he would probably never use recommendation set again. So how to support high precise prediction is an important issue that we must take into consideration when we design the recommendation engine.

In our research we focus on two problems: (1) After the website organizer modifies the content of the website, the regular patterns that are found off-line no longer have the same precision in predicting the next browsing behavior. The recommendation engine must use other methods to generate other recommendation candidates for the current user. (2) The recommendation engine must guarantee that it can immediately detect the new trend of user behavior, adapt its candidate set through a real-time operation, and put them in a list according to the decending order of importance for the current user. In Fig. 2, we propose a newly recommendation engine architecture that can support dynamic adapting recommendation.

There are three major components in the recommendation engine: site topology matrix, recommendation generator, and recommendation coordinator. The first component is a site topology matrix that keeps the information about site structure. The second component, recommendation generator, uses three methods to generate recommendations. These three methods are the FM model and PPED, the Click-Count Matrix and the Hyperlink Weight Matrix, and the Most Frequent Browsing Web Page. Recommendation coordinator is the third part of the architecture, which calculates and ranks the weight of each web page in the recommendation set. The ordered recommendation set, the output of recommendation coordinator, includes a few ordered recommendations whose weight is higher than the threshold standard. Only the web pages whose information strength is higher than the threshold could be held in the ordered recommendation set. Therefore the recommendation engine can use these three components to generate real-time and appropriate recommendations for each on-line user.

### 3.1. Site Topology Matrix

First of all, the site topology matrix represents information about the structure of the current website, ex. Fig. 3. The website's structure is constructed by the hyperlinks between its

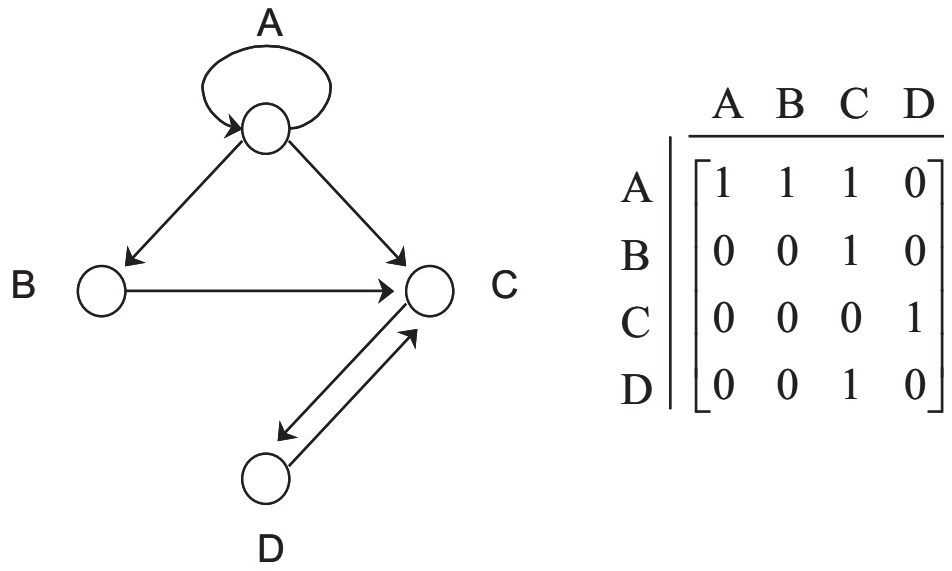


**Figure 3. A Simple Website Topology and Its Site Topology Matrix**

web pages. Once the content of this website is changed, the site topology matrix must be reconstructed as well in order to include the newest information of the web structure. The site topology matrix, like an adjacency matrix, uses 1 to represent that a hyperlink exists between two web pages and 0 to represent that no hyperlinks exist between them. Therefore, the matrix could represent whether there is a hyperlink between two successive browsing web pages and thus makes it possible to have a quick access to site structure information. The matrix is especially useful for the path completion process, as it needs to justify the relationship between two steps. It is easy to construct the matrix; all we need is a simple crawler program.

Many web usage mining methods use a special string to represent the structure of the website. Some problems arise, however. For example, the string does not always provide the whole picture, it is hard to find information, and the access performance is low. It is advisable to keep in mind that the information of the website structure is processed by the computer, not people. That is why that we must arrange to have site topology represented from the perspective of the machine. In case the number of web pages is large-so much so that a large matrix is formed-we can reduce the size of the topology matrix by partitioning a session into transactions of different web page classes. Using transaction analysis to replace session analysis will reduce the problem

domain effectively.

### 3.2. Recommendation Generator

The recommendation generator uses three methods to generate recommendations. First, the FM models represent users' distinct browsing behavior off-line. These FM models are put into clusters by similarity measurement. Each cluster collects the same or similar behavior patterns from anonymous users. Then PPED algorithm does partial matching between the on-line user's short-term behavior and discovered clusters. The best matching clusters is used to predict the next browsing web page. This method relies primarily on long-term trends to generate recommendations, though it also takes the on-line user's short-term clickstream into account by means of partial matching. We can therefore use the last browsing steps of each on-line user to justify which cluster the current user's behavior belongs to and then effectively generate a prediction about the next browsing target.

The second recommendation method uses both the click-count matrix and the hyperlink weight matrix, as illustrated in Fig. 4. The method is the key point of our research that satisfies the requirement of dynamically adaptive recommendation. Both matrices map directly to the site topology matrix. The click-count matrix stores the number of times each hyperlink is clicked by the user. The content of each cell in the hyperlink weight matrix represents the importance of this hyperlink. The large value it holds, the more click frequency the hyperlink has. Once the user clicks a hyperlink on web page, then the relative location of this hyperlink in the click-count matrix is increased to 1, a presupposition that there exists a hyperlink in the site topology matrix. If a user browses a web page but there appears to be no hyperlink between the current and the previous web pages, it means that the on-line user accepts a recommendation from the recommendation set and this fact will be processed by the recommendation-count matrix in the recommendation coordinator. There are four kinds of time intervals in the click-count matrix: short-term, middle-term, long-term, and history. Different time interval matrices help to capture all major and minor variations in the user's browsing behavior.

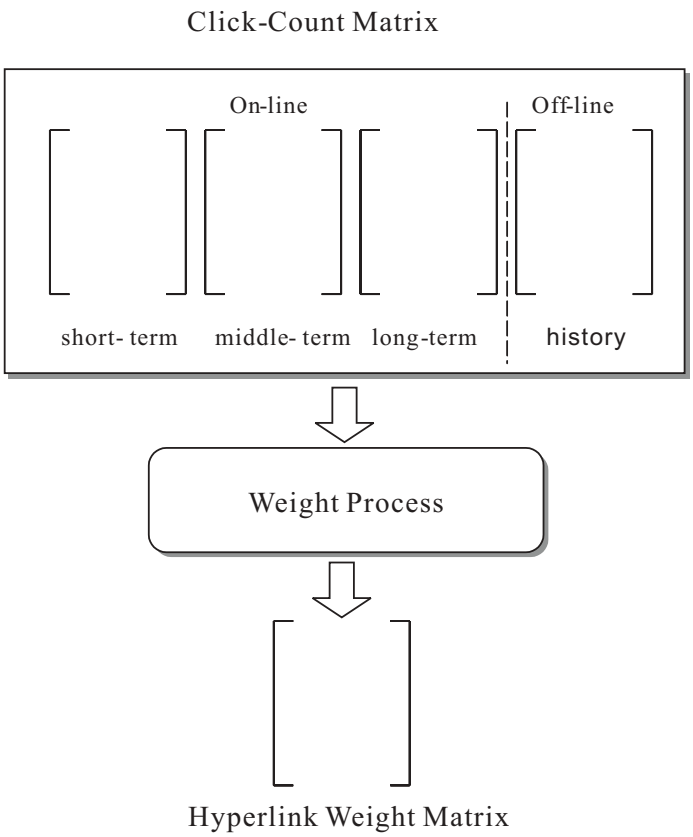


**Figure 4. Click-Count Matrix versus Hyperlink Weight Matrix**

**Table 1. Weight Adaptive Table**

| Id | S vs M | M vs L | L vs H | Adjust |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | W- |
| 1 | 0 | 0 | 1 | X |
| 2 | 0 | 1 | 0 | X |
| 3 | 0 | 1 | 1 | W+ |
| 4 | 1 | 0 | 0 | W+ |
| 5 | 1 | 0 | 1 | W++ |
| 6 | 1 | 1 | 0 | W++ |
| 7 | 1 | 1 | 1 | W+++ |
| S: Short-Term, M: Mid-Term, L: Long-Term, H: History | | | | |
| 0: Descend, 1: Ascend | | | | |
| -: Decrease, +: Increase, W: Weight, X: No Operation | | | | |

The content of the click-count matrices needs to be adjusted based on two kinds of statistical time: on-line and off-line. In the on-line mode, there are three click-count matrices: short-term, middle-term, and long-term, depending on the clickstream of on-line users and the content they modify. In the off-line mode, the click-count matrix is updated from the web access log in the

off-line operation of web usage mining. The ratios between these four matrices are used to modify the content of the hyperlink-count matrix. Therefore the hyperlink weight matrix uses the ratios of click-count matrices and adjusts its content based on Table 1.

The third recommendation method does nothing but count the click times of each web page. We can use the statistical results to justify which web page has been browsed most frequently. Note that the default page, which is also the first browsing web page of all websites, must be left out from the candidate set. In general, that page is unlikely to be what users want to browse next. The second-level web pages are often ignored too, but the decision rests with the site organizer.

### 3.3. Recommendation Coordinator

The third component of the recommendation engine is a recommendation coordinator, as illustrated in Fig. 5. The coordinator recalculates the weight of each candidate suggested by the recommendation generator. Because it is possible for the recommendation generator to duplicate the results of different methods, all recommendations must multiply their relative weight that forms the cells of the recommendation weight matrix: Weight.FM, Weight.Hyperlink, and Weight.Frequency. These ratios represent the accuracy of each recommendation generation method. Once a candidate is duplicated by a different method, its weight must be recalculated by adding up the values of all of its appearances. In other words, it is the summation of each candidate that forms the real weight of each web page. After the recalculating process, the recommendation coordinator filters all of the candidates with an appropriate threshold. In this last step, the recommendation coordinator sorts the candidates by the weight that results from the search engine. The recommendation coordinator has four Recommendation-Count Matrices, which are used to detect the acceptance ratio of the three recommendation generation methods. The weight process uses the same matrices to adapt to the content of the Recommendation Weight Matrix.

The ordered recommendation set for the on-line user, arranged in order of importance, is

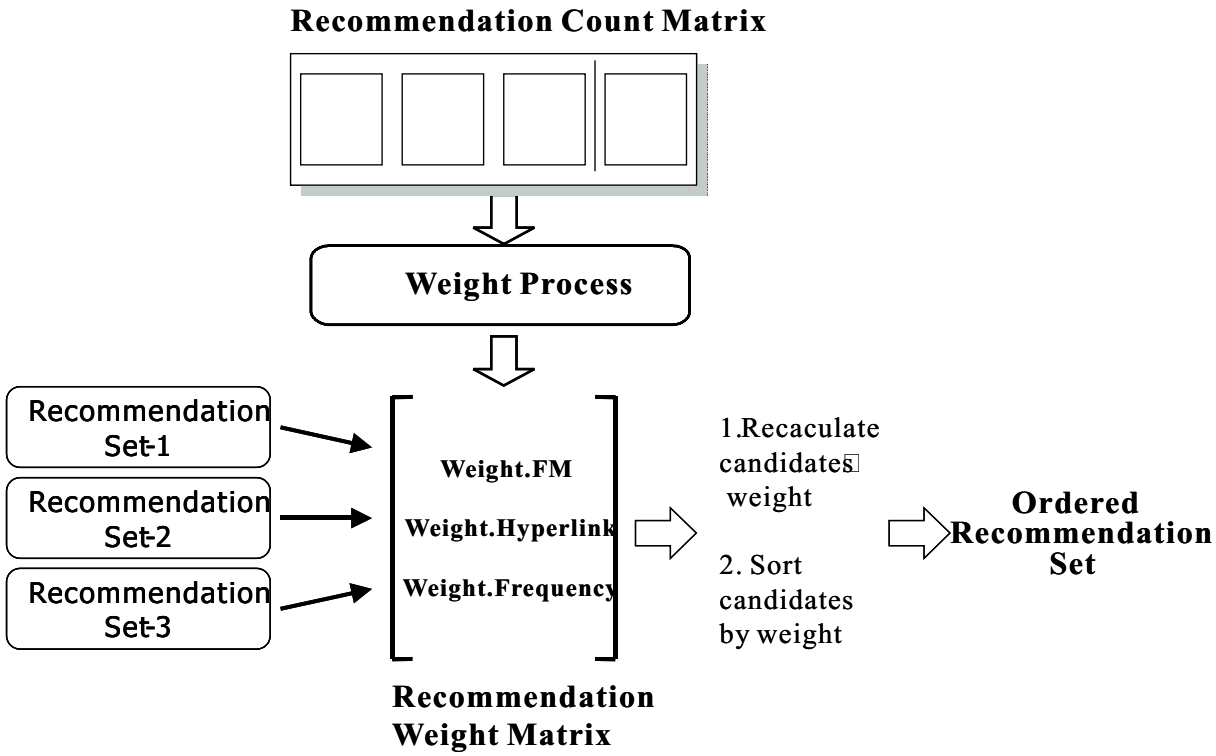


**Figure 5. The Process of Recommendation Coordinator**

dependent on the short-term clickstream of on-line user as much as on anything else.

## 4. Experimental Result

In our experiment we used the access log from the web site of the Department of Electrical Engineering and Institute of Microelectronics at the National Cheng Kung University (www.ee.ncku.edu.tw). The test data contain 14,566 requests issued between 2002/2/18 and 2002/2/27 from 1,296 different IP addresses. With the session window size set at 4, the log file produces 1,708 sessions with session length longer than 6. These longer sessions ensure better pattern-recognition and help generate a larger number of meaningful clusters. The short patterns are ignored in the clustering operation, so there are no recommendations for short sessions. There are 280 web pages in total on the web site.

In this section, first, we check the Click-Count Matrix to adjust the weight of each hyperlink to different adapt rates and time interval ratios (short vs. middle term, middle vs. long term, and

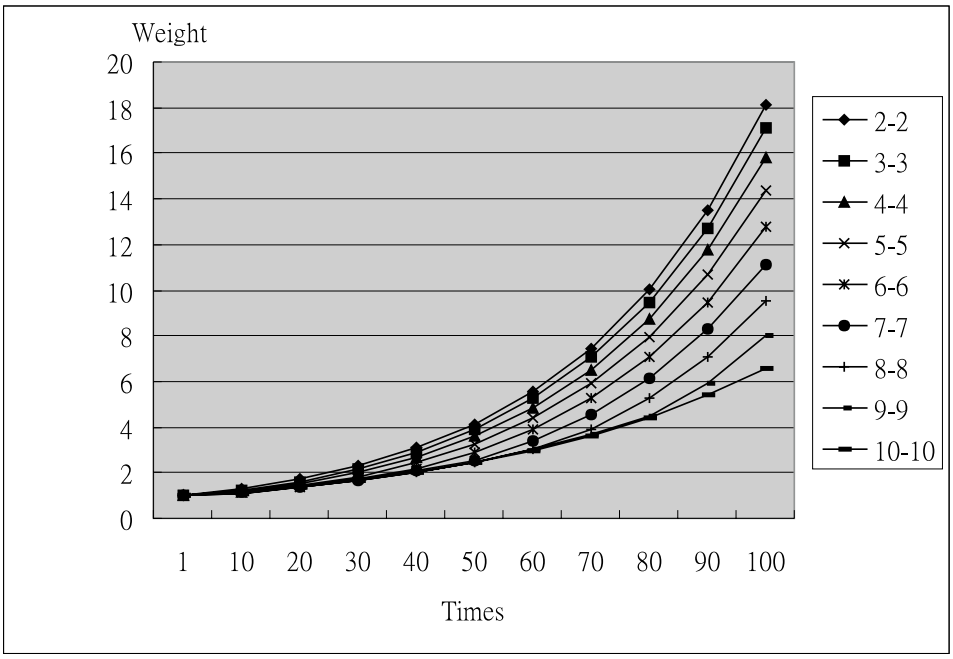


**Figure 6. The results under different time ratios-the short-, middle-, and long-term click-count matrices-with the adapt rate 0.01.**

long term vs. a history of various click-count matrices). Second, we illustrate the precision of our recommendation engine. At the end, we discuss the experiment results and give suggestions for future work.

In Figure 6, suppose the appearance of a hyperlink increasing gradually, then the recommendation engine can use the click-count matrices to detect the phenomenon and adjust the weight accordingly. Each hyperlink's weight is closely related to the adapt rate and the ratios between the short-, middle-, and long-term click-count matrices as described in Table 1. In Figure 6 we can clearly find that the different ratios of short, middle, and long terms generate different weight results with the adapt rate 0.01. Once a hyperlink is detected to have a higher frequency of use than before, its weight will be adjusted immediately.

If we use different adapt rates to adjust the weight of each of the hyperlinks whose frequency of appearance is still increasing, then results are as shown in Figure 7. The web site organizer can decide which adapt rate to use based on the characteristics of that particular web site. For example, if the web site is a news site which needs to recommend hot news to its browser, then

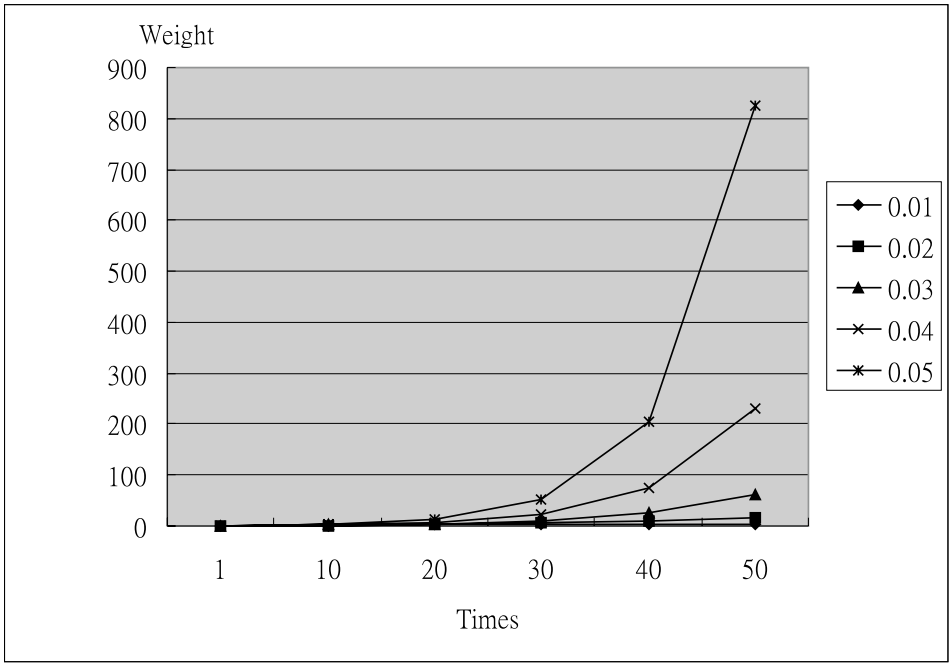


**Figure 7. The results of using a variety of adapt rates.**

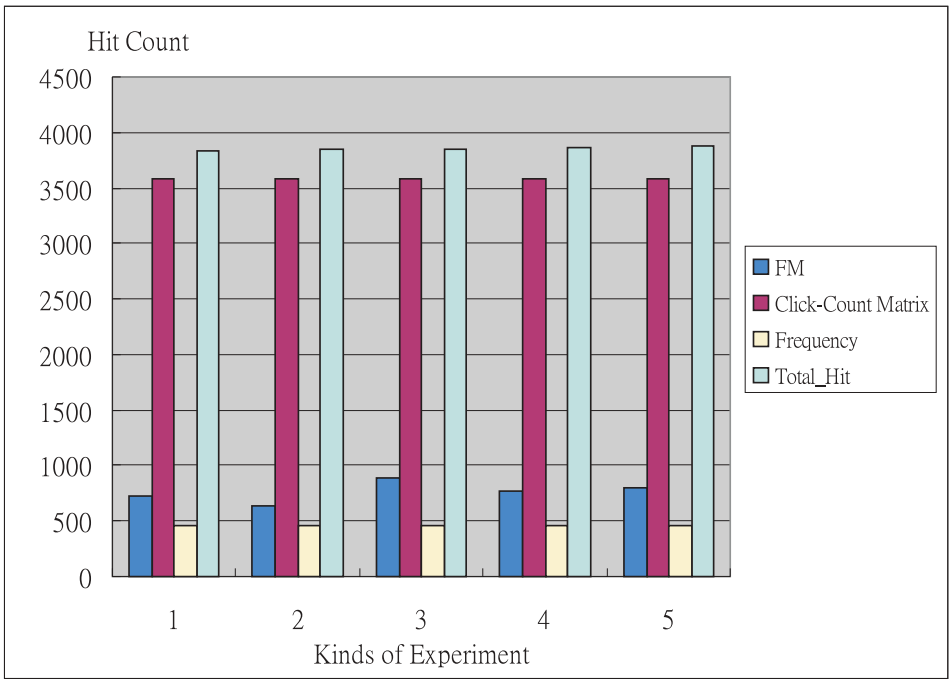


**Figure 8. The simulation results of different similarity, from 0.01 to 0.05.**

it must use an adapt rate of more bits to make the focal points of the hot news stand out as soon as possible.

Figure 8 shows the simulation results of different similarity clusters, from 0.01 to 0.05.

The recommendation engine uses three methods whivh were described in section 3 to generate recommendations for online browsers. The three methods were described in section 3. FM models users' behavior and then proceeds to do similarity clustering, which helps keep track of long term browsing trends. Both Click-Count Matrices and Hyperlink Weight Matrix detect the short term trend online, which is especially useful when web site structure is updated. The last method, the most frequently browsed web pages, is the easiest recommendation-giving method. In Figure 8, the Click-Count Matrix and Hyperlink Weight Matrix attain higher precision than the other methods. The hit rate of the FM method is low. Based on our observation, we believe that it is due to many short sessions and spiders' searching records in log file. It is not easy to establish a behavioral pattern because its supports might not be greater than the threshold to form a cluster.

Future work to improve on the recommendation engine includes the following tasks: <1> to find a better similarity-recognizing method to speed up the clustering operation, <2> to focus on shorter sessions to improve precision, <3> to looking for a better performance algorithm that has a good hit rate and works at a higher speed.

## 5. Conclusion

In this paper, we have proposed a new architecture for recommendation engine with both short-term trend detection and dynamically adaptive recommendations. We used PPED, a partial match algorithm, to speed up the similarity measurement between the on-line user's latest browsing steps and off-line discovered clusters. In addition, we enhanced the accuracy of the recommendation sets by employing the following methods: (1) click-count matrices and a hyperlink weight matrix capture the short-term trends from on-line users' browsing behavior; (2) a site topology matrix provides a quick access to full website structure information; (3) a recommendation weight matrix and recommendation count matrices adapt the acceptance rate of each recommendation generating method to respond to emerging trends on-line; (4) all the three methods mentioned above are used concurrently to increase the accuracy of recommendation.

To optimize the efficiency of a recommendation engine, further research needs to be conducted. First and foremost, the information sources of users' behavior have to be increased. Because the web access log is designed for website debugging, a lot of information can not be retrieved from it. For instance, we do not have any information regarding which items are taken from the cart in e-commerce. Information from application server log, in this case, will come in handy and thus should be added to the data pool. Secondly, some load balance software must be used to speed up recommendation generation, an operation that is usually extremely time-consuming and thus suggests poor performance.

## References


[1] M. S. Chen, J. Han, and P. S. Yu, "Data Mining: An Overview from Database Perspective," *IEEE Trans. on Knowledge and Data Engineering*, vol. 8, no. 6, pp. 866–883, Dec. 1996.

[2] URL http://docs.yahoo.com/docs/pr/release634.html.

[3] R. Cooley, B. Mobasher, and J. Srivastava, "Web Mining: Information and Pattern Discovery on the World Wide Web," *In Proceedings of ICTAIf97*, Nov. 1997.

[4] R. Cooley, B. Mobasher, and J. Srivastava, "Data preparation for mining World Wide Web browsing patterns," *Journal of Knowledge and Information Systems*, vol. 1, no. 1, pp. 5–32, Feb. 1999.

[5] C. Shahabi, F. Banaei-Kashani, J. Faruque, and A. Faisal, "FeatureMatrices: A Model for Efficient and Anonymous Web Usage Mining," *In Proc. of EC-Web 2001*, Sept. 2001.

[6] Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules," *In Proceedings of the 20th VLDB*, Sept. 1994.

[7] J. S. Park, M. S. Chen, and P. S. Yu, "Using a Hash-Based Method with Transaction Trimming for Mining Association Rules," *IEEE Transactions on Knowledge and Data Engineering*, vol. 9, no. 5, pp. 209–221, Nov. 1998.

[8] I. Cadez, D. Heckerman, P. S. C. Meek, and S. White, "Visualization of Navigation Patterns on Web Site Using Model Based Clustering," *In Proceedings of the sixth ACM SIGKDD*, August 2000.

[9] A. Joshi and R. Krishnapuram, "Robust Fuzzy Clustering Methods to Support Web Mining," *In Proc. of SIGMODf98 Workshop on Data Mining and Knowledge Discovery*, June 1998.

[10] Y.-J. SU, H. C. Jiau, and S.-R. Tsai, "Design a Dynamically Adaptive Recommendation Engine on WWW," *In Proc. of NCS 2001, Taiwan*, Dec. 2001.