dimension vector space and then use one of the efficient hierarchical clustering methods to convert original web transactions into clusters where a cosine function is used as a similarity metric. So far the task of DPM can be viewed as a similarity-matching problem between a DT and a number of web transaction clusters for finding the most similar cluster or clusters to the DT. The final house-cleaning step is to find suitable pages (hyperlinks) from the result cluster(s) for promoting to the web user of the DT. We depict the DPWP structure as Figure 1.
The paper is organized as follows. In section 2, we explore how the DPM mechanism works. In section 3, we discuss how to effectively clustering web logs. Finally, we conclude with related work.

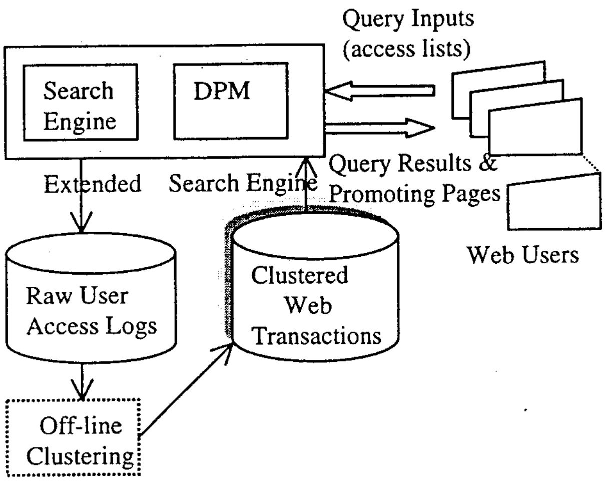


Figure 1: Dynamic Personalized Web Promoting (DPWP) Framework

2. DYNAMIC PROMOTING MECHANISM (DPM)

We assume that there is a clustered web transaction from web logs available. We design a DPM mechanism to solve the problem of identifying the most similar pages to an ongoing web user. First we define a DT to represent the current behavior of a web user, i.e. dynamic user trend. Second, we match a DT with a group of clusters to find most relevant cluster or clusters to a web user. The DPM mechanism is working as an agent to discover pages most relevant to web users. These pages are expected not only to respond the shifting interests of the user from one domain to another, but also promote the high-interest pages to the web users based on the knowledge of the clustered transactions (extracted from historical user access logs) In this section, we first present the process of extract DTs from a access list and then explore the problem of DPM and propose our solution.

2.1 The Dynamic Transaction (DT)

When we explore the first subproblem of the personalized web – how to effectively interpret the dynamic user trends, two obvious approaches are readily brought to mind: first, to find the relevant pages to the last visited page; or second, to find the relevant pages based on the page and frequency pairs in an access list of a user session. We, however, propose an alternative more efficient strategy. First, if we make use of the last accessed page only, we would probably have skipped the other pages accessed within the access list and overlooked other implicated patterns hidden in the overall access list. Second, if we use all the pairs of page and frequency within an access list, although we have seemingly taken all the access records into consideration, we question the efficiency of doing so, since some of the access pages are meaninglessly traversed. In [3], they have also discussed a similar problem. They recommend that only real access patterns can be meaningfully mined. We find an interesting implication from their research results. That is, *whenever a user accesses a new page (i.e. no matter whether it is a forward reference or a backward reference), there must exist a maximum forward reference corresponding to the last accessed page.* In other words, the combination of pages in a maximum forward reference path can meaningfully represent the interest trend of a web user over time. We define a *maximum forward reference* as a *Dynamic Transaction (DT).*
As an example, assume the access list, <ABCBDBAGH>, within a user session. This access list can be represented as a corresponding traversing tree shown in Figure 2. This reveals that during different time periods, T1 to T9, the user might move forward or backward to form a traversing tree. Based on the definition of DT given in the previous paragraph, we may extract distinct DTs from the corresponding traversing tree. Some of the transactions are duplicate ones because the user might move to and from a node (i.e. dangling) in a same path of the traversing tree. If we keep tracing as in Figure2, when in T1, we get a DT of <A> because node A is the sole accessed node at this stage; when in T2, the DT becomes <AB> because a forward reference was further taken by the user; at T3, we get <ABC>; at T4, we still get a DT of <ABC> because one backward reference was taken by the user. We cannot judge the user intention at this moment (i.e. at T4), so we set the same DT when the user dangles around (moves to and from) at the same path in the tree. At T5, however, we get an another distinct DT of <ABD> because the user clearly moves to a new path in the traversing tree. It implies that the user change interests or preference in her access list, which suggests giving up the nodes belonging to a different old path (DT). When in T6 and T7, we still obtain a DT of <ABD> for the same reason as at T4. In T8, we get a new DT <AG> because of the user changing access path again. At T9, we get a DT of <AGH> owing to a forward reference made by the user. The DTs extracting process is illustrated in Figure 3 and we also summarize the DT generating process in Table 1.

1998 International Computer Symposium
Workshop on Computer Networks, Internet, and Multimedia
December 17-19, 1998, N.C.K.U., Tainan, Taiwan, R.O.C.

A NEW AGENT-ORIENTED APPROACH FOR PERSONALIZED WEB MANAGEMENT

Yuh-Chi Lin and Paul Hadingham

Department of Computer Science, The University of Western Australia,
Crawley, WA 6907, Australia
Email: {yuh,paul}@cs.uwa.edu.au

ABSTRACT

Finding techniques for constructing a personalized web is an important issue in the World Wide Web research. In this paper, we explore the problem of dynamical personalized web promotion – how to effectively interpret dynamic user trends and how to efficiently promote the most relevant pages to users. First, we present a simple idea based on dynamic transactions to effectively represent the changing trends of web users over time. Second, we formalize the dynamical personalized web promotion problem as a similarity matching between a dynamic transaction and a group of clustered web logs. The dynamic promoting mechanism is designed for this purpose where two algorithms, cluster identification and page identification, are also presented. Finally, we introduce our method for efficiently clustering web logs. Parts of the ideas above have been implemented, and the preliminary results are encouraging.

1. INTRODUCTION

In real life, a share investor might surf through the web sites to look up online financial news. Suppose he first reads an article in CNN; and then reads another article from Morgan Stanley's home page; and then jumps to the pages of New York Times; and then goes back to CNN's article; and then jumps to one of the articles in Washington Post; and finally visits Wall Street Journal's article. As these articles are provided by different information providers, and these referenced pages may not be linked together directly, the reader must visit many intervening pages before she can locate the interesting ones. Potentially, there may be a personalized web for distinct users. That is, we may categorize web users into different groups. Whenever a user accesses a certain page, the pages relevant to the accessed page will be highlighted in some way to suit the personal needs of web users. In general, we may identify the categories of users by analyzing user access logs.

In relation to personalized web site construction research, two major approaches have been undertaken. The first approach is to set up dynamic personalized web sites. Such a web site dynamically adjusts its presentation to suit the particular web user based on her profile. For example, the AVANTI project fits into this approach [7]. Technically, such web sites must hold distinct web user profiles and provide corresponding web topologies of hyperlinks. However, this method usually effects the actual structure of a web site and the reorganization process may be time-consuming or labour-intensive. The second method is the agent-oriented approach. Within such a web site, an agent acts as an intelligent tour guide. Whenever a web user visits a page, this tour guide (i.e. agent) should be able to instantly highlight the user the most relevant pages, where the WebWatcher is one of the cases in such a category [2].

In this paper, we focus on the second approach. Here are two sub-problems to construct an intelligent tour-guide web site. First, how to effectively interpret the dynamic user trends. Second, how to efficiently determine the most relevant pages to a web user. Aiming at the first case, if the interests of a web user are misinterpreted, the sought pages are bound to dissatisfy the user [3,4,5,10,11,17]. As to the second case, if the assistant agent takes too long to search for relevant pages, such an agent still will not be counted as a successful one.

To tackle the previously described problem, we propose a solution which we call Dynamic Personalized Web Promoting (DPWP) illustrated in Figure 1. In the DPWP framework, we first define a naive Dynamic Transaction (DT) to effectively identify the dynamic interests of web users. Then, we propose a Dynamic Promoting Mechanism (DPM) to find relevant pages to web users. For brevity, in the DPM mechanism, we match the pre-defined DT with the web logs to get interesting pages for highlighting to the ongoing users. According to the related literature [16], we know that it is less efficient to find useful results from raw web logs when they are unstructured and it is also inefficient to try to extract anything from such a large sized log files online. Therefore, we adopt a procedure of web log clustering as follows: First, we cleanse web logs into transactions (i.e. maximum forward reference paths which will be further discussed in section 2 and 4); second, we remove redundant transactions to avoid too many unnecessary initial clusters at the very beginning of clustering; third, we regard each transaction as a vector in a multi-

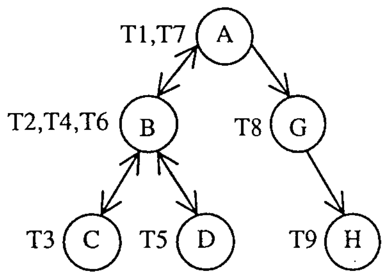


Figure 2: The traversing tree corresponding to a user access list <ABCBDBAGH>

Table 1: A generating process of dynamic transactions

Time Period	Accessed Page	Dynamic Transactions	Access Types
T1	A	A	forward access
T2	B	AB	forward access
T3	C	ABC	forward access
T4	B	ABC	backward access
T5	D	ABD	forward access
T6	B	ABD	backward access
T7	A	ABD	backward access
T8	G	AG	forward access
T9	H	AGH	forward access

Notes: 1. The forward access means web users move to a new node or move forward to another new access path. 2.The backward access implies that web users move backward to previously traversed node in the existing path.

2.2 Algorithms for Dynamic Promoting Mechanism

As we discussed in section 1. We formalize the DPM problem as a similarity-matching problem between a DT and a number of web transaction clusters. Two algorithms, Similarity Identification (SI) and Page Identification (SI), are required. SI is used to find the most similar cluster or clusters to a determined DT. The function of PI is to search the most promising pages from the result cluster(s) for the web user of the DT. In Figure 4, we give a brief diagram of DPM.

2.2.1 *Algorithm for Similarity Identification (SI)*

Once a DT for an ongoing web user is generated and given a database, Dc, containing all clustered web transactions extracted from web logs, we now begin identifying the most similar cluster(s) to DT. A cluster C

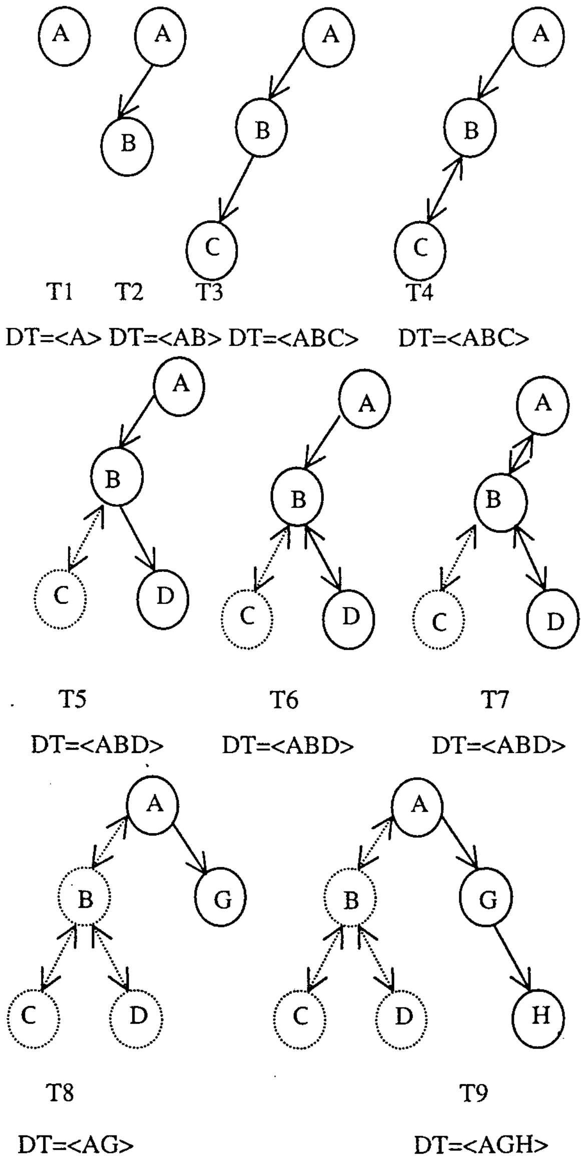


Figure 3: A Process of Extracting Dynamic Transactions

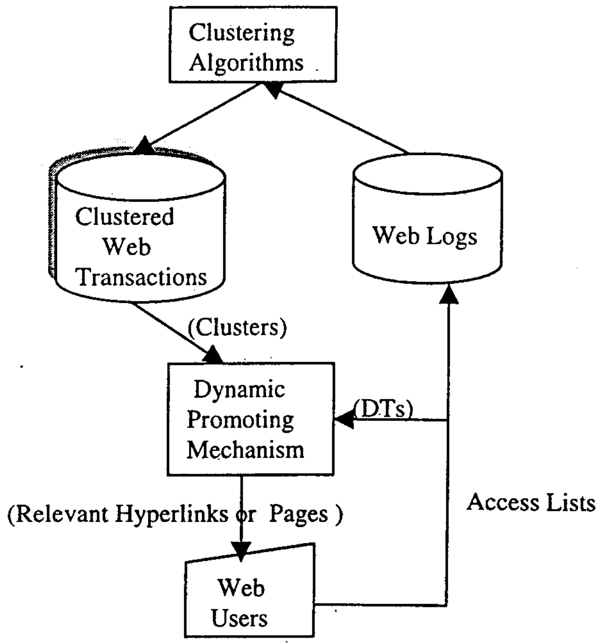

Clustering Algorithms
Clustered Web Transactions
Web Logs
(Clusters)
Dynamic Promoting Mechanism
(DTs)
(Relevant Hyperlinks or Pages)
Access Lists
Web Users

Figure 4: Diagram for Dynamic Promoting Mechanism (DPM)

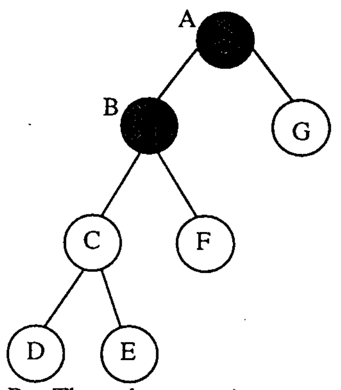
is defined to be a most similar cluster to DT if there exists a maximum cosine between DT and any clusters in Dc. We use the following Group Average Similarity (GAS) function to define the similarity between DT and clusters:

$$\mathrm{GAS(DT,C)}= \sum_{tran_i \in C} \frac{1}{|C|} \mathrm{Sim(DT}, tran_i).$$

where $\mathrm{Sim(DT}, tran_i)$ is the average cosine of the pairwise similarities between a DT and each transaction in a cluster C. Therefore, after having computed all the GAS of each clusters in the database Dc, at least one cluster with minimum GAS value would be selected as the most similar clusters to DT. The process of identifying such clusters is called similarity identification.

2.2.2 *Algorithm for Page Identification (PI)*

Recall that a web transaction cluster includes a number of maximum forward reference paths (i.e. transactions). Therefore, in a clustered web transaction database Dc, each cluster includes a number of web transactions, which are defined as non-redundant paths. As we defined in section 2.1, a DT also makes use of maximum forward reference to represent the trend of a web user over time. Whenever the most similar cluster(s) is (are) found, we can begin the final house-cleaning task as follows: first, we can construct a forest of n-ary graphs. Second, we check if the DT exists in the forest. Finally, if we find a DT then we get the neighboring pages nearest to the DT. For example, given a cluster C {<T_1, ABCD>,<T_2, ABCE>,<T_3, ABF>,<T_4, AG>} with 4 paths and a current DT of <AB> extracted from an ongoing user access list <ACDCAB......>, the task is to find the most similar pages to DT. According to the previous procedure, we first construct a forest corresponding to the cluster C (see Figure 5); second, we locate the DT's position as the solid nodes in the following graph; and third, if we suppose pages C,F,D,E,G are candidates to be promoted to the ongoing web user under the predefined threshold (i.e. the number of pages to be promoted), pages C ,G, and F might be promoted with highest priority because these nodes are only one move (step) away, while D and E are at least two-steps distant from the DT of an ongoing user. To consider the distance (by move or by steps), we count the number of steps from each pages nodes of a DT. We can easily cast this problem to a subgraph searching problem in a forest of graphs [6].



DT= <AB>. The web transactions are merged into a graph. Nodes C, F and G with one-step distance to B.
Nodes D and E with 2 steps to B.

Figure 5: An Example for Pages Identification (PI)

3. WEB LOG CLUSTERING

According to our previous exploration, it is evidently difficult to directly find useful results from raw web logs when they are unstructured and it is also inefficient to try to extract anything from such log files because of their size. Clustering techniques have been explored in a number of research papers [1,8,9,12,13,18]. Therefore, we can adopt a procedure of web log clustering to organize the user access patterns. In previous sections, we speak of using maximum forward reference to represent an atomic web transaction before clustering, and then we can match the dynamic transactions of users to discover relevant pages from these clusters. In other words, we aim at constructing a group of clusters of maximal forward reference to suit the clusters of our DPM mechanism, which was previously explored in section 2. Now we can adopt the clustering procedure as follows: First, we cleanse web logs into transactions (i.e. maximum forward reference paths as discussed in section

2); second, we remove redundant transactions to avoid too many unnecessary initial clusters; third, we regard each transaction as a vector in a multi-dimension vector space and then use one of the efficient hierarchical clustering methods to convert original web transactions into clusters where a cosine function is used as a similarity metric. Below, we give a number of definitions related to our web transaction-clustering problem. Further, in subsequent section we show the clustering steps for web logs.

3.1 Representation of Web Transactions

User access logs (Web Logs or Logs): The log files are automatically updated each time a request for a resource reaches the web server. They customarily contain the domain name (or IP address) of the request, the user name of the user who generated the request (if applicable), the date and time of the request, the method of the request (GET or POST), the name of the file requested, the result of the request (success, failure, error, etc.), the size of the data sent back, the URL of the referring page, and the identification of the client agent.

Nodes or vertices: In this paper, nodes or vertices are used interchangeably. They denote objects traversed back or forth in accordance with the hyperlinks and icons provided by a web environment.

Backward reference nodes: Nodes, which might be revisited because of their location, rather than content. For example, in a www environment, to search a sibling node a user may usually use backward icon and then forward searching, instead of opening a new URL address. So, a backward reference pattern implies that a previously traversed object occurs in the same user transaction.

Forward reference nodes: Opposite to backward reference nodes, the forward reference nodes are the nodes, which are first traversed in a transaction in appearance sequence.

Maximal forward references (Paths or Web Transactions or Dynamic Transactions): In a transaction, when backward references occur, a forward reference path terminates. This resulting forward reference path is termed a maximal forward reference. Such a maximal forward reference is initially proposed by [3]. The maximal forward references are referred to as paths in this paper. We define a path as an atomic web transaction after web logs are cleansed. That is, we would use paths for web log clustering purpose. The schema of web transaction t is in the form of <TID,d_1d_2, ... ,d_i, ..., d_n>, where TID represents transaction identifier; d_i is a sequence of traversal nodes ;and $1 \le i \le n$.

3.2 Clustering Algorithms

Before we discuss how to cluster web transactions, we abstract the problem as follows: given a non-redundant web transactions database, each web transaction is represented in terms of the schema <TID,(d_1,d_2,...,d_i,...d_n>, where TID represents a transaction identifier; d_i is the node accessed by web users; $1 \le i \le n$; the clustering is to determine a set of cluster in which inter-cluster similarity is minimized and intra-cluster similarity is maximized.

The above web transaction clustering problem can be formulated as the document clustering problem. In our problem, in each user transaction there is a number of hyperlinks similar to words in the documents clustering problem. Our clustering method begins by placing each user transaction into a distinct cluster. Pairwise similarities between all such clusters are computed, and the two closest clusters are then merged into a new cluster. The algorithm starts with a set of singleton clusters each containing a single web transaction. These singleton clusters are the clusters at the bottom of the clustering tree, called leaves. Starting with the singleton clusters the algorithm identifies pairs of clusters that are most similar and merges them into a single cluster. The process is iterated until only a single cluster, the root, is left. When merging two clusters their intra-similarity is calculated. The singleton clusters have an intra-cluster similarity of 100% since they contain only one transaction. The binary tree, as depicted in Figure 6, constructed during the clustering process, contains the complete clustering information including all inter- and intra-cluster similarity of the first common cluster. The notion of similarity between web transactions is crucial to obtain highquality groupings of information. For our clustering problem, we can define a group average similarity to estimate the pairwise similarity between two clusters C and C'. The group average similarity function is as the following.

$$\mathrm{GAS}(C, C') = \sum_{tran_i, tran_j \in C, C'} \frac{1}{|C|} \times \frac{1}{|C'|} \mathrm{Sim}(tran_i, tran_j).$$

Briefly, an $O(n^2)$ time algorithm for our problem proceeds as follows:

1. Determine and store the similarity between each pair of clusters. Initially, each cluster only has one single web transaction.
2. Determine the pair of clusters with the highest similarity between them and agglomerate them.
3. Update the pairwise similarity and the new nearest neighbours.
4. If more than one cluster still exists go to Step2, (or if less than the threshold of slices go to Step2)

This algorithm can be performed in $O(n^2)$ time, since Step one requires $O(n^2)$ time and Steps 2-4 are performed n – 1 times and they can be completed in O(n) time. Step 3 requires determining the new nearest neighbour for clusters that had one of the nearest neighbour, but in the single link case we are guaranteed that the new nearest neighbour will be the new agglomerated cluster [13].

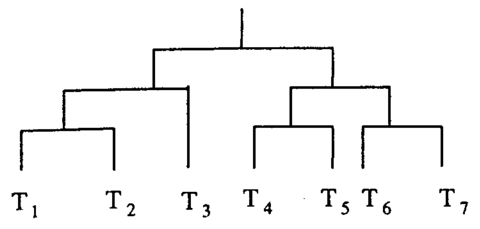


Figure 6: A dendgrogram shows how the transaction clusters are merged hierachichially.
(Note that T_i is represented as the format of $<TID_i, d_{i1}, d_{i2}, \ldots, d_{in}>$, where d is a web transaction.)

4. EXPERIMENTS AND DISCUSSION

To validate our conjecture that DPM mechanism can both efficiently and effectively find relevant pages for web users, we carried out a number of experiments. In our experiments, we first observed the web page structure and then analyzed the user access patterns of the web logs of an institutional sub-organization. Then, we made a synthetic cleansed web log for our experimental purpose. Suppose we assume that the number of distinct nodes is 400 in a web log from which we generated 3000 access lists. The average length of each access list is 20 (i.e. nodes). Each access list could generate 6 maximum forward reference paths on average. That is, we could get around 18000 maximum forward reference paths. However, after eliminating the redundant (i.e. frequently traversed) paths, there are about 14600 non-redundant transactions before clustering. According to the assumption we gave in section 1, the clustering procedure should be executed offline, so we omitted the time of clustering here. We have also clustered the 14600 non-redundant transactions into a different number of clusters for comparison purpose.
The preliminary results show that the DPM method is feasible for locating relevant nodes for an ongoing web user. It only took a couple of seconds to finish the computation even on a 200 MHz personal computer with 16 megabytes of RAM. In general, our approach is highly likely to efficiently find the results in real time. We also discovered that if the number of clusters was set too large, the DPM method might find a result with a different number of nodes. However, when we set the size of clustering more than a certain level (i.e. the number of clusters is good enough), we could obtain near consistent nodes in the final result. We infer that there might be an optimized clustering level to suit our problem. Therefore, in the future work, we would further demonstrate the effectiveness and efficiency of our methods when they are incorporated into a real system (i.e. a search engine).

5. RELATED WORK

The personalized web idea is a new research field in the context of the WWW. Such an endeavour tries to establish a new gateway for bridging the gap between the two groups (i.e. information consumers and providers). The idea of the personalized web is only an abstract goal. Web mining and other related techniques have contributed a number of useful concrete techniques in the personal web environment. Therefore, there are numerous unexpected challenges to be conquered for building a truly satisfactory web.
Various approaches have been taken to handle web personalization. In one approach [7,15], criteria for reconfiguration is provided by the web page designers. The linkage among web pages is adapted in a predefined way based on human knowledge or skills. WebWatcher [2] presented a machine learning approach to extract traverse suggestions. In Yan et al.'s research [16], an on-line matching approach is first proposed. They inherit the approach of adding "suggestions" to a requested page in a piggyback way from [2]. Yan et al. make use of the knowledge from clustering web logs for categorizing users. In [14], they propose an approach of synthesizing relevant web pages into an index page. This approach is also derived from [2].
Our approach mainly expands the research of [3], the WebWatcher project of [2], and Yan et al.'s work in [16]. One of Chen et al.'s contributions is to show how to derive maximum forward reference paths from a noisy user access list within a user session. We borrow the idea of maximum forward reference path to normalize the dynamic behaviour of web users and we also apply the approach to help cleanse user access logs before they are clustered. Our proposal based on dynamic transactions of web users and web log clustering using the maximum forward reference path as an atomic web transaction has not been explored before.

6. CONCLUSION

The development of the WWW has surpassed the levels we could have imagined a few years back. The current phase of the WWW appears as an immense global information supermarket, which contains diverse information products. As the information supermarket environment (i.e. WWW) becomes more complex, the correlation between production and consumption becomes more asynchronous. That is, the information customers find difficulty in traversing the supermarket and the information providers worry about how to efficiently promote their products.
In this paper, our most important contributions are: first, we propose a straightforward approach to identify and

interpret the dynamic consuming (i.e. traversing) behaviours of web users; second, we provide a new web clustering approach using user access paths; third, we make use of an existing web clustering technique for locating the natural groups hidden in the web logs as a knowledge base for efficiently discovering the pages relevant to an ongoing web user. A DPWP framework that integrates the key components also is given to describe the global view of how to construct a dynamic personalized web. According to our preliminary experimental results, our methods represent a feasible solution for constructing a more intelligent personalized web.

7. REFERENCES

[1] Agrawal, Rahesh et al. (1998) Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications. Proc. of the ACM SIGMOD Int'l Conference.

[2] Armstrong, Robert et al. (1995) WebWatcher: A Learning Apprentice for the World Wide Web. In Working Notes of the AAAI Spring Symposium: Information Gathering from Heterogeneous, Distributed Environments, pp. 6-12. (also in http://www.cs.smu.edu/afs/cs.cmu.edu/project/theo-6/web-agent/www/project-home.html).

[3] Chen, M.S. et al. (1996) Data Mining for Path Traversal Patterns in a Web Environment. In Proc. of the 16th International Conference on Distributed Computing Systems, pp 385-392.

[4] Cooley, R. et al..(1997a) Grouping Wen Page References into Transactions for Mining World Wide Web Browsing Patterns. Technical Report TR 97-021, University of Minnesota, Department of Computer Science, Minneapolis.

[5] Cooley, R. et al. (1997b) Web Mining: Information and Pattern Discovery on the World Wide Web. In Proc. of the 9th IEEE Int'l Conference on Tools with Artificial Intelligence (ICTAI 97).

[6] Even, Shimon (1979) Graph Algorithms. Computer Science Press.

[7] Fink, J. et al. (1996) User-oriented adaptivity and adaptability in the avanti project. (in http://zeus.gmd.de/projects/avanti.html).

[8] Han, Eui-Hong et al. (1997) Clustering Based on Association Rule Hypergraphs. In Proc. of Data Mining and Knowledge Discovery (DMKD97).

[9] Koller, Daphne and Sahami, Mehran (1997) Hierarchically Classifying Documents Using Very Few Words. Proc. of the Int'l Conference Of Machine Learning (ICML-97), pp.170 –178.

[10] Moore, Jerome. et al. (1997) Web Page Categorization and Feature Selection Using Association Rule and Principal Component Clustering. In Proc. Of 7th Workshop on Information Technologies and Systems (WITS 97).

[11] Mobasher, Bamshad. et al. (1996) Web Mining: Pattern Discovery from World Wide Web Transactions. Technical Report TR 96-050, university of Minnesoda, Department of Computer Science, Minneapolis.

[12] Nasraoui, Olfa et al. (1997) Mining Web Access Logs Using Relational Competitive Fuzzy Clustering.

[13] Olson, Clark F. (1995) Parallel Algorithms for Hierarchical Clustering. Parallel Computing, 21(8): pp. 1313-1325.

[14] Perkowitz, Mike and Etzioni, Oren (1998) Adaptive Web Sites: Automatically Synthesizing Web Pages. Proc. of AAAAI98.

[15] Scotts, P. and Furuta. (1991) Dynamic Adaptation of Hypertext Structure. Proc. of 3rd ACM conference on Hypertext.

[16] Yan, Tak Woon et al. (1996) From User Access Patterns to Dynamic Hypertext Linking. Technical Report, Hewlett-Packard Laboratories,1996.

[17] Wexelblat, Alan (1998) History-Rich Tools for Social Navigation. CHI'98 Summary, ACM Press. (Also http://wex.www.media.mit.edu/people/Footprints2/fp-v2.html)

[18] Zhang, Tian et al. (1996) BIRCH: An Efficient Data Clustering Method for Very Large Dtabase. Proc. of ACM SIGMOD Conference, pp. 103-114.