Discovering Web Access Patterns and Trends by Applying OLAP and Data Mining Technology on Web Logs*

Osmar R. Zaïane Man Xin Jiawei Han

Virtual-U Research Laboratory and Intelligent Database Systems Research Laboratory
School of Computing Science
Simon Fraser University
Burnaby, BC, Canada V5A 1S6
E-mail: {zaiane, cxin, han}@cs.sfu.ca

Abstract

As a confluence of data mining and WWW technologies, it is now possible to perform data mining on web log records collected from the Internet web page access history. The behaviour of the web page readers is imprinted in the web server log files. Analyzing and exploring regularities in this behaviour can improve system performance, enhance the quality and delivery of Internet information services to the end user, and identify population of potential customers for electronic commerce. Thus, by observing people using collections of data, data mining can bring considerable contribution to digital library designers.

In a joint effort between the TeleLearning-NCE project on Virtual University *and NCE-IRIS project on* data mining, *we have been developing the knowledge discovery tool,* WebLogMiner, *for mining web server log files. This paper presents the design of the* WebLogMiner, *reports the current progress, and outlines the future work in this direction.*

1 Introduction

Web servers register a (web) log entry for every single access they get in which they save the URL requested, the IP address from which the request originated, and a timestamp. With the rapid progress of World-Wide Web (WWW) technology, and the ever growing popularity of the WWW, a huge number of Web access log records are being collected. Popular web sites can see their web log growing by hundreds of megabytes every day. Condensing these colossal files of raw web log data in order to retrieve significant and useful information is a nontrivial task. It is not easy to perform systematic analysis on such a huge amount of data and therefore, most institutions have not been able to make effective use of web access history for server performance enhancement, system design improvement, or customer targeting in electronic commerce. However, many people have realized the potential usage of such data.

Using web log files, studies have been conducted on analyzing system performance, improving system design, understanding the nature of web traffic, and understanding user reaction and motivation[7, 9, 18, 19]. One innovative study has proposed adaptive sites: web sites that improve themselves by learning from user access patterns[14]. While it is encouraging and exciting to see the various potential applications of web log file analysis, it is important to know that the success of such applications depends on what and how much valid and reliable knowledge one can discover from the large raw log data.

Currently, there are more than 30 commercially available applications for web log analysis and many more are available free on the Internet (The university of Illinois maintains a list of web access analyzers on a HyperNews page accessible at http://union.ncsa.uiuc.edu/HyperNews/get/www/log-analyzers.html). Regardless of their price, most of them are disliked by their users and considered too slow, inflexible and difficult to maintain [17]. We can even argue that most of these web log analysis tools are very limited in the results they can provide. After reviewing over twenty such analysis tools, including Getstats, Analog, Microsoft Intersé Market Focus, and WebTrends, we found that most of the reports of these tools reveal only frequency count and/or low conceptual descriptive level. While these types of report are valuable, they are certainly not enough. The most frequent reports pre-defined by web log analysis tools are: a summary report of hits and bytes transferred, a list of top requested URLs, a list of top referers, a list of the most common browsers used, hits per hour/day/week/month reports, hits per domain report, an error report, a directory tree report, etc. Despite the fact that some of the reports can be customized with some of these tools, the majority of the currently available web log analysis tools have rigid pre-defined reports. Most if not all of these web log analysis tools have limitations with regard to the size of the web log files, whether it is physical in size or practical in time because of the low speed of the analysis. To reduce the size of the log files to analyze,

*Research is supported in part by the Natural Sciences and Engineering Research Council of Canada, TeleLearning-NCE (note: NCE stands for Canadian Networks of Centres of Excellence) and NCE/IRIS-2.

web log analysis tools make assumptions in order to filter out some data like failed requests (i.e. errors) or page graphic requests, or to round off the log entries by combining similar requests. Different assumptions are made for each of the web log analysis tools resulting in the prospect of different statistics with the same log file. It has been reported in [2] that the analysis of the same web log with different web log analysis tools ended up with different statistic results.

Overall, the current web log analysis tools are still limited in their performance, the comprehensiveness and depth of their analyses, and the validity and reliability of their results.

The recent progress and development of data mining and data warehousing has made available powerful data mining and data warehousing systems [6, 4]. Many successful data mining systems can handle very large data files like the web log files. However, we have not seen a systematic study and development of data warehousing and mining systems for mining knowledge from web access log records. Recent research and development of data mining technology have promoted some studies on efficient data mining for user access patterns in distributed systems, referred to as mining path traversal patterns [5]. Understanding user access patterns in a web site using these mining techniques not only helps improve web system design, but also leads to wise marketing decisions (e.g. putting advertisements in proper places, classifying users, etc.). However, as mentioned in [4], mining path traversal patterns is still in its infancy.

In this paper we propose to use data mining and data warehousing techniques to analyze web log records. Based on our experience on the development of relational database and data warehouse-based data mining system, DBMiner [10], by the Intelligent Database Systems Research Laboratory at Simon Fraser University, and on the development of a Web-based collaborative teaching and learning environment, Virtual-U, by the Virtual-U Research Laboratory at Simon Fraser University, we jointly study the challenging issues on data mining in web log databases, and propose the WebLogMiner system which performs data mining on web log records collected from web page access history.

The remaining of this paper is organized as follows. In Section 2, we describe the design of a data mining system for web log records. Our implementation efforts and experiments are presented in Section 3. Finally, Section 4 summarizes our conclusions and future enhancements.

2 Design of a Web log miner

The most commonly used method to evaluate access to web resources or user interest in resources is by counting page accesses or "hits". As we will see, this is not sufficient and often not correct. Web server log files of current common web servers contain insufficient data upon which thorough analysis can be performed. However, they contain useful data from which a well-designed data mining system can discover beneficial information.

Web server log files 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.

A log entry is automatically added each time a request for a resource reaches the web server. While this may reflect the actual use of the resources on a site, it does not record reader behaviours like frequent backtracking or frequent reloading of the same resource when the resource is cached by the browser or a proxy. A cache would store resources and hand them to a client requesting them without leaving a trace in the log files. Frequent backtracking and reload may suggest a deficient design of the site navigation, which can be very informative for a site designer, however, this cannot be measured solely from the server logs. Many have suggested other means of data gathering like client-site log files collected by the browser, or a Java Applet. While these techniques solve problems created by page backtracking and proxy caching, they necessitate the user's collaboration, which is not always available. Until the web server log files are enriched with more collected data, our data mining process is solely based on the information currently gathered by the web servers.

Researchers working on web log analysis discredit the use of web access counts as indicators of user interest or measure of the interestingness of a web page. Indeed as described by Fuller and deGraaff in [7], access counts, when considered alone, can be misleading metrics. For example, if one must go through a sequence of documents to reach a desired document, all documents leading to the final one get their counters incremented even if the user is not interested in them at all. The access counters alone do not account for the user's ability to access the information and the appropriateness of the information to the user. Nonetheless, when access counts are used in conjunction with other metrics, they can help infer interesting findings.

Despite the impoverished state of the server logs, much useful information can be discovered when using data mining techniques. The date and time collected for each successive request can give interesting clues regarding the user interest by evaluating the time spent by users on each resource, and can allow time sequence analysis using different time values: minutes, hours, days, months, years, etc. The domain name collected can allow practical classification of the resources based on countries or type of domain (commercial, education, government, etc.). The sequence of requests can help predict next requests or popular requests for given days, and thus help improve the network traffic by caching those resources, or by allowing clustering of resources in a site based on user motivation. Notwithstanding, the server logs cannot be used as recorded by the web server and need to be filtered before data mining can be applied.

In the WebLogMiner project, the data collected in the web logs goes through four stages. In the first stage, the data is filtered to remove irrelevant information and a relational database is created containing the meaningful remaining data. This database facilitates information extraction and data summarization based on individual attributes like user, resource, user's locality, day, etc. In the second stage, a data cube is constructed using the available dimensions. On-line analytical processing (OLAP) is used in the third stage to drill-down, roll-up, slice and dice in the web log data cube. Finally, in the fourth stage, data mining techniques are put to use with the data cube to predict, classify, and discover interesting correlations.

2.1 Database construction from server log files: data cleaning and data transformation

The data filtering step is a typical step adopted by many web log analysis tools. While typically web log analysis tools may filter out requests for page graphics (as well as sound and video) in order to concentrate on data pertaining to actual page hits, we tend to keep these entries because we believe they can give us interesting clues regarding web site structure, traffic performance, as well as user motivation. Moreover, one user action can generate multiple server requests, and some of them are requests for page media. Some of these requests are important to deduce the intended action of the user. Another typical cleaning process consists of eliminating log entries generated by web agents like web spiders, indexers, link checkers, or other intelligent agents that pre-fetch pages for caching purposes. We chose not to screen out these requests generated by the web agents. It is often interesting and useful to analyze web agents' behaviour on a site and compare the traffic generated by these automated agents with the rest of the traffic. The data filtering we adopted mainly transforms the data into a more meaningful representation. We tend to consider most of the data are relevant and eliminate a minimal amount of data.

There are two types of data cleaning and data transformation, one that does not necessitate knowledge about the resources at the site and one that does. *Cleaning* the date and time field of the log entry, for instance, does not need any knowledge about the site itself. The date and time field is simply restructured in a set of fields to specify the day, month, year, hour, minute and seconds. Filtering out server requests that failed or transforming server error codes is also generic. Transforming IP addresses to domain names is independent from the site content as well. However, associating a server request or a set of server requests to an intended action or event clearly necessitates knowledge about the site structure. Moreover, different dynamically generated web pages can be the result of a single script, thus, an identical server request. Knowledge about the parameters provided to the script to generate the dynamic page, or knowledge about the necessary sequence in the request history before a request for a script, can be essential in disambiguating a server request and associating it to an event.

Metadata provided by the site designers is required for the knowledge-based data cleaning and transformation. The metadata consists of a mapping table between a server request (URL) with parameters, if available, or a sequence of requests (URLs) and an event with a representative URL. The transformation process replaces the request sequence by the representative URL and adds the event tag to the log entry.

After the cleaning and transformation of the web log entries, the web log is loaded into a relational database and new implicit data, like the time spent by event, is calculated. The time spent by event (or page) is approximated from the difference between the time the page for the current event is requested and the time the next page is requested with an upper-bound threshold for the case when the user does not come back to the same server. This notion of time spent is an approximation of the actual perusal duration since it intrinsically includes the time for network transfer, navigation inside the page, etc. It may seem a biased metric but can be very useful comparing pages with the same design.

2.2 Multi-dimensional web log data cube construction and manipulation

After the data has been cleaned and transformed, a multi-dimensional array structure, called a data cube, is built to aggregate the hit counts. The multi-dimensional data cube has numerous dimensions (i.e. generally more than 3), each dimension representing a field with all possible values described by attributes. For example, the dimension *URL* may have the attributes: *server domain, directory, file name* and *extension*, or the dimension *time* may have the attributes: *second , minute, hour, day, week, month, quarter, year*. Attributes of a dimension may be related by partial order indicating a hierarchical relationship among the dimension attributes. Hierarchies are generally pre-defined, but in some cases partitioning the dimension in ranges automatically generates these hierarchies. For example the dimension file size can be partitioned into size ranges and later grouped into categories like: *tiny, small, medium, large, huge*.

An n-dimensional data cube, $C[A_1, \ldots, A_n]$, is an n-D database where $A_1, \ldots, A_n$ are n dimensions. Each dimension of the cube, A_i, represents an attribute and contains $|A_i| + 1$ rows where $|A_i|$ is the number of distinct values in the dimension A_i. The first $|A_i|$ rows are *data rows*. Each distinct value of A_i takes one data row. The last row, the *sum* row, is used to store the summation of the counts of the corresponding columns of the above rows. A data cell in the cube, $C[a_{1,i_1}, .., a_{n,i_n}]$, stores the *count* of the corresponding (generalized) tuple of the initial relation, $r(A_1 = a_{1,i_1}, .., A_n = a_{n,i_n}, count)$. In the case of the web log cube, *count* is for resource hits. A sum cell in the cube, such as $C[sum, a_{2,i_2}, .., a_{n,i_n}]$, where sum is often represented by $*$ or a keyword all, stores sum_1, the *sum of the counts* of the generalized tuples which share the same values for all the second to the n-th columns, i.e., $r(*, A_2 = a_{2,i_2}, .., A_n = a_{n,i_n}, sum_1)$. Conceptually, a data cube can be viewed as a lattice of *cuboids*. The *n-D space* (i.e., *base cuboid*) consists of all data cells (i.e, no $*$'s in any dimension). The *(n-1)-D space* consists of all the cells with a single $*$ in any dimension, such as $r(*, a_{2,i_2}, .., a_{n,i_n}, sum_1)$, and so on. Finally, the *0-D space* consists of one cell with n $*$'s in all the n dimensions, i.e., $r(*, *, \ldots, *, sum_n)$.

A 3-D data cube is shown in Fig. 1. Notice since

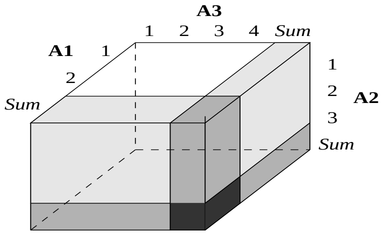

Figure 1: A 3-D data cube with summary layers

a large number of the cells in the cube can be empty, to handle sparse cubes efficiently, sparse matrix technology should be applied in data cube construction [13, 20]. There have been many methods proposed recently for efficient computation of data cubes, such as [1, 20], which is beyond the scope of this study.

Examples of dimensions in our web log data cube include the following. Notice that each dimension is defined on a concept hierarchy to facilitate generalization and specialization along the dimension.

- URL of the resource, where the concept hierarchy used is defined on the server directory structure;
- type of resource, defined on a pre-built file type hierarchy;
- size of the resource, defined on a range hierarchy;
- time at which the resource was requested, defined on a time hierarchy;
- time spent in page, defined on a range hierarchy of seconds;
- domain name from which the request originated, defined on a pre-built domain hierarchy;
- agent that made the request, defined on a pre-built hierarchy of known web agents and browsers;
- user, defined on a pre-built user hierarchy;
- server status, defined on an error code hierarchy.

The multi-dimensional structure of the data cube provides remarkable flexibility to manipulate the data and view it from different perspectives. The sum cells allow quick summarization at different levels of the concept hierarchies defined on the dimension attributes.

Building this web log data cube allows the application of OLAP (On-Line Analytical Processing) operations, such as drill-down, roll-up, slice and dice, to view and analyze the web log data from different angles, derive ratios and compute measures across many dimensions.

The drill-down operation navigates from generalized data to more details, or specializes an attribute by stepping down the aggregation hierarchy. For example, presenting the number of hits grouped *by day* from the number of hits grouped *by month*, is a drill-down along the hierarchy time. The roll-up is the reverse operation of the drill-down. It navigates from specific to general, or generalizes an attribute by climbing up the aggregation hierarchy. For example, the aggregation of total requests from *group-by organization by day* to *group-by country by day* is a roll-up by summarization over the server domain hierarchy.

The slice operation defines a sub-cube by performing a selection on one dimension by selecting one or some values in a dimension. It is a literal cut of a slice (or slices) on the same dimension. For example, the selection domain = ".edu" on the dimension server domain, is a slice on the educational internet domain. The dice operation is a set of consecutive slice operations on several dimensions. It defines a sub-cube by performing selections on several dimensions. For example, a sub-cube can be derived by dicing the web log data cube on four dimensions using the following clause, country = "Canada" and month = 11/97 and agent = "Mozila" and file_type = "cgi". These OLAP operations assist in interactive and quick retrieval of 2D and 3D cross-tables and chartable data from the web log data cube which allow quick querying and analysis of very large web access history files.

2.3 Data mining on web log data cube and web log database

On-line analytical processing and the data cube structure offer analytical modeling capabilities, including a calculation engine for deriving various statistics, and a highly interactive and powerful data retrieval and analysis environment. It is possible to use this environment to discover implicit knowledge in the web log database by implementing data mining techniques on the web log data cube. The knowledge that can be discovered is represented in the form of rules, tables, charts, graphs, and other visual presentation forms for characterizing, comparing, associating, predicting, or classifying data from the web access log.

These data mining functions are briefly explained as follows.

- Data characterization: This function characterizes data in the web log. It consists of finding rules that summarize general characteristics of a set of user-defined data. The rules are generated from a generalized data cube produced using the web log data cube and the OLAP operations. For example, the traffic on a web server for a given type of media in a particular time of day can be summarized by a characteristic rule.
- Class comparison: Comparison plays the role of examining the Web log data to discover discriminant rules, which summarize the features that distinguish the data in the target class from that in the contrasting classes. For example, to compare requests from two different web browsers (or two web robots), a discriminant rule summarizes the features that discriminate one agent from the other, like time, file type, etc.
- Association: This function mines association rules (in the form of "$A_1 \wedge \cdots \wedge A_i \rightarrow B_1 \wedge \cdots \wedge B_j$") at multiple-levels of abstraction. For example, one

may discover the patterns that accesses to different resources consistently occurring together, or accesses from a particular place occurring at regular times.

- Prediction: Prediction involves predicting values or value distributions of an attribute of interest based on its relevance to other attributes. Both relevance analysis and predictive model construction need statistical analysis techniques. This helps prediction of possible values of missing data or the value distribution of certain attributes in a set of objects. For example, the access to a new resource on a given day can be predicted based on accesses to similar old resources on similar days, or the traffic for a given page can be predicted based on the distribution of traffic on other pages in the server directory.

- Classification: Classification consists of building a model for each given class based upon features in the web log data and generating classification rules from such models. The models are constructed by analyzing a training web log data set whose class label is known. The classification rules can be used to develop a better understanding of each class in the web log database, and perhaps restructure a web site or customize answers to requests (i.e. quality of service) based on classes of requests.

- Time-series analysis: Time-series analysis is to analyze data collected along time sequences to discover time-related interesting patterns, characteristics, trends, similarities, differences, periodicity, and so on. It may also involve attribute relevance analysis, model construction, classification, and prediction. Thus time-series analysis explores most of the techniques developed in the above (data mining functions) plus its own techniques for time-sequence search, similarity analysis, periodicity analysis, and so on. For example, time-series analysis of the web log data may disclose the patterns and trends of web page accesses in the last year and suggest the improvement of services of the web server.

Since most data mining functions other than time-series analysis share many commonalities with the work in traditional data mining systems, such as IBM Intelligent Miner (Quest) [1], Silicon Graphics MineSet [3], DBMiner [10], this paper will not discuss the application of these data mining functions to web log mining in detail. Our focus will be on time-series analysis because web log records are highly time-related, and the goals of data mining with web log records are largely aimed at mining time-related patterns.

Our time-series analysis includes *network traffic analysis, event sequence and user behavior pattern analysis, transition analysis*, and *trend analysis*. With the availability of data cube technology, such analysis can be performed systematically in the sense that analysis can be performed on multiple dimensions and at multiple granularities. Moreover, there are major differences in time-series analysis of web log mining in comparison with other traditional data mining processes.

We take trend analysis as an example to illustrate such a process. In the analysis of the trend of web accessing in the Virtual-U environment, we would like to see how a user changes his/her web navigation behaviour and focuses his/her attention to interested topics. The analysis takes the following steps.

1. data/user selection. Since a Virtual-U user accesses the Virtual-U web pages regularly, those who access the Virtual-U web page only occasionally for curiosity will not be included in the analysis. That is, access regularity will be taken as a standard (or threshold) to filter out scattered accesses.

2. cycle detection. For web accessing, a regular Virtual-U user usually starts a Virtual-U session, traverses a set of pages with possible inputs to some pages, and then leaves the Virtual-U sessions for a while or for a long time before coming back to start another session. Thus the starting or restarting of a Virtual-U web page, following by a sequence of other local web page accesses forms a cycle. A data mining task needs to detect such cycles effectively for meaningful analysis. We have developed techniques on how to find such cycles and detect periodicity efficiently using data cube structure and OLAP techniques [8].

3. trend analysis. With the accumulation of the discovered sequences and periods, analysis can be performed on them to discover patterns and trends with different interests. One kind of patterns which can be easily discovered is the repetition of similar web page accesses. For the same set of web pages, many users will access such pages repeatedly in a similar pattern. Number of accesses will be associated with each page node to register such access frequency.

   Moreover, with data cube technology, one can roll-up the access history to get general web page accessing statistics. Such access statistics form a group of web access trees/graphs. These access trees demonstrate some clear trend along the time axis: taking the structural complexity of the access tree as a focus, one can easily see the trend is the gradually reduced structural complexity of the access tree, which shows more experienced users are more focused on specific topics with reduced number of random searches to other scattered topics.

   Notice with the data cube technology, such a trend can be demonstrated with different time granularities, such as by week, bi-week, month, bi-month, quarter, etc. by simple clicking the button of time axis. However, the trend will be "discovered" by a human with the help of visualization tools instead of a fully automated process. The detailed process of this web page access trend analysis is presented in Example 3.5.

3 Experiments with the web log miner design

Based on our design elaborated in Section 2, experiments have been carried out to analyze a server-based online teaching/learning system, Virtual-U, which is developed by the Virtual-U Research Laboratory at Simon Fraser University. The system enables customized design, delivery, and enhancement of education and training courses delivered over the World Wide Web (WWW). Virtual-U is a collaborative teaching and learning environment which accumulates large collections of multimedia resources.

Virtual-U consists of six different major components as follows: (1) V-Groups which is a web-based conferencing system, (2) course structuring tools, (3) system administration tools, (4) a grade book, (5) upload and submission tools, and (6) a virtual workspace.

V-Groups is a predominant component in which the on-line discussion and collaboration take place. It holds and organizes sets of conferences for different courses. The instructors and students read and write messages in conferences. This amasses a large number of messages and generates heavy access traffic. Thus, we will base our examples in this paper primarily on V-Groups events.

SFU-based Virtual-U server currently hosts different fieldsites ranging from academic institutions to industries across Canada.

The goal of the analysis of the server log files is to understand how the system is used in terms of the usage patterns and user behavior patterns, which will give us insights of the system design, user interface design, and the characteristics of different user populations.

3.1 Data cleaning and transformation

Before we could feed the information from the log file into a relational database, we had to first clean and transform the raw data.

The original log file entries include the following information: the domain name (or IP address) of the request, the user login id, the date and time of the request, the method of the request (GET or POST), the location and the name of the file requested, the result of the request (success, failure, error, etc.) and the size of the data transferred. An example entry is shown as in Table 1, where the user_id is masked by a faked id, such as user_1.

There were two processes we went through in the data cleaning and transformation. First, all entries of the log files were mapped one on one into a relational database. In other words, no information was lost in this process. Entries that recorded server request failures, authentication failures, page graphics, etc. were all kept in the database. We believe all entries have potential useful information depending on the purpose of one's analysis and the kind of questions one wants to answer. Once we get the maximum information into the relational database, we can always draw the relevant information and leave out the rest according to the analysis needs.

The transformed data entries had one additional field, the *fieldsite*, indicating which one of the seven fieldsites hosted by Virtual-U server the request was from. This field was derived from the path of the file requested. Example entries are given as shown in Table 2.

The second process was to infer user actions (events) from patterns of requests. In this process, we combined, eliminated and/or directly mapped the same set of log entries and translated them into a set of user actions. As Virtual-U is an interactive environment, most pages were generated dynamically by a user's requests of certain cgi scripts. After a page was generated, there were a number of options available for the user to choose from. Once the user made his/her choice, new cgi script(s) were called and a new page was generated possibly with another set of options. To infer user actions from the requested cgi scripts and corresponding parameters and then tag them with plain English was the fundmental step for further in-depth analysis of user behavior.

There were several general problems associated with this data process which would occur in the similar interactive online system. These problems and their solutions are described as follows.

First, extraneous information was mixed with useful information. Log entries that recorded the request failures, page graphics, etc. were irrelevant to our current analysis. To solve this problem we simply defined those entries and then eliminated them.

Second, multiple server requests were generated by the same user action. Two cases were associated with this problem. First, the user action could be identified by a single entry of the entry set. In this case, we simply tagged that entry and discarded the rest. Second, the user action had to be identified by multiple entries of the entry set. In this case, we had to infer the user action by processing multiple entries.

Third, multiple user actions may generate the same server request. For example, in the case a user adds a new message to a conference, or preview a to-be-added message, or cancel addition of a message, all three actions call the same cgi script (VG_addmsg.cgi) and generate the same entries in the log except the time stamps. Fortunately, for most cases like these in this study, we managed to solve the problem by examine the context in which the entry occurred. In the case we could not identify clearly, we translated the entry with multiple tags. In association with such problem, a suggestion made to the system developers is to use different scripts for different user actions or use same script with parameter(s) signed with different values.

Fourth, local activities (browser functions, such as backing and forwarding) were not recorded. While there is no easy and quick solution to this problem at present stage, certain backtracking can be inferred in some cases from the user behavior sequences. Frequent backtracking in the same context can indicate certain user interface problem(s).

After the second process, the transformed data entries had one more attribute — user action (event). Example entries are given as in Table 3.

One point of interest was the duration of the event. It gives additional information of the interaction between events. However, time-based metrics are subject to a fair amount of noise because of reasons such

vanc01m03-115.bctel.ca - user1 [10/Apr/1997:19:06:44 -0700]	"GET /SFU/cgi-bin/VG/dspcnf.cgi?ci=40050&fp=MLIST&query= HTTP/1.0" 200 3927
cnts4p14.uwaterloo.ca - user2 [10/Apr/1997:19:09:43 -0700]	"GET /Waterloo/cgi-bin/UI/UI_welcome.cgi HTTP/1.0" 200 1288

Table 1: A fragment of the original log file.

requester IP address	user_id	date	time	request	fieldsite	files requested	result	size received
vanc01m03-115.bctel.ca	user1	10/Apr/1997	19:06:21	GET	SFU	/SFU/cgi-bin/VG/vgstart.cgi	200	386
vanc01m03-115.bctel.ca	user1	10/Apr/1997	19:06:34	GET	SFU	/SFU/cgi-bin/VG/VGTP_ListConfs.cgi?st=all_joined	200	9294
cnts4p14.uwaterloo.ca	user2	10/Apr/1997	19:09:43	GET	Waterloo	/Waterloo/cgi-bin/UI/UI_welcome.cgi	200	1288

Table 2: A fragment of the transformed log file.

as time spent on file transfer, increasing use of "offline" browsers and unknown user behavior between events [18]. Nevertheless, when such metrics are used selectively on certain events and with valid sample size, they can still reveal very useful information [15].

To get the duration of the event, we used the time stamp of current entry as the (event) *on-time* and the time stamp of the next entry as the (event) *off-time*.

3.2 Multi-dimensional data cube construction and manipulation

After data cleaning and transformation, a relational database was constructed with the processed data, and a multi-dimensional data cube was constructed. In addition to the dimensions given in Section 2.2, two more dimensions were added to the data cube for this experiment: one is the *fieldsite* dimension defined on a pre-built education and industry institution hierarchy, and the other is the *event* dimension defined by a set of user actions that could take place in the system environment, organized in a partial order.

Once the multi-dimensional data cube is constructed, various OLAP techniques are applied to provide further insight of any target data set from different perspectives and at different conceptual levels. This counts as summarization function in data mining as well.

Some typical summarization includes the following:

- *request summary* to give request statistics for all pages/files;
- *domain summary* to give request statistics from different domain;
- *event summary* to give statistics of the occurring of all events;
- *session summary* to give statistics of session;
- *bandwidth summary* to give statistics of generated network traffic;
- *error summary* to give statistics of all the error messages;
- *referring organization summary* to give statistics of where the users were from;
- *browser summary* to give statistics of the use of different browsers.

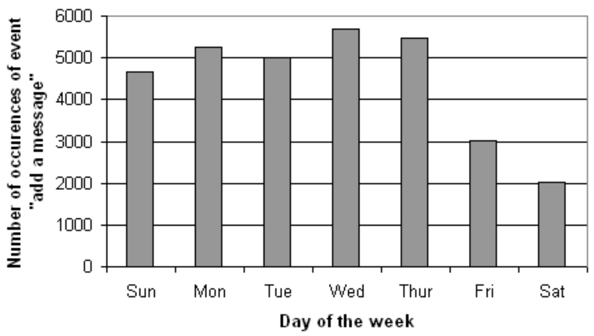

Figure 2: OLAP analysis of Web log database.

Such summaries could be applied to any given user population, conference(s), course(s) or fieldsite(s) and for any given length of time, hourly, daily, weekly, monthly and semesterwise. From the relational database and data warehouse point of view, these summaries correspond to *group-bys* on different dimensions, multiple abstraction levels, and their arbitrary combinations. Dicing, slicing, and drilling can be performed on the Web log data cube to examine request statistics, network traffic information, user information, etc. conveniently. From such results, we could gain insights of the system usability, accessibility and general performance as well as insights of the behavior characteristics of different user populations.

Example 3.1 (OLAP analysis of Web log) Web log databases can be analyzed by OLAP analysis techniques. Here we illustrate a simple yet typical example of summarization using OLAP technique. For example, we were interested in finding out how many new messages were created on each day of the week over a whole semester for a particular fieldsite. In this analysis, we chose a particular value of interest (i.e. slice) on the *event* dimension, example "Use V-Groups", then focus on a particular level (i.e. drill-down in the same dimension) ("add a message"), and choose a particular value of interest on the *fieldsite* dimension, we then projected the intersect of both dimensions on the *time* dimension to locate the cells we wanted. Figure 3 shows the result. Using OLAP technique, such result can be generated interactively using the data cube structure.

...	fieldsite	files requested	user action	result	size received
...	SFU	/SFU/cgi-bin/VG/vgstart.cgi	start V-Groups	200	386
...	SFU	/SFU/cgi-bin/VG/VGTP_ListConfs.cgi?st=all_joined	list all joined conferences	200	9294
...	Waterloo	UI_welcome.cgi	load welcome page	200	1288

Table 3: A fragment of the log file with user action added.

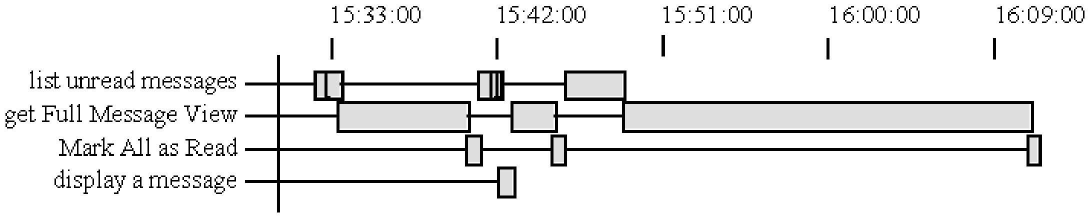

Figure 3: Usage pattern

From Figure 3 one can see the distribution of messages created on days of the week. When similar analyses performed on other events and/or other fieldsite using the same time dimension, we could do interesting comparisons among the results to gain further knowledge of the analysis domain. □

From these summarizations one can ask questions such as:

- Which components or features are the most/least used?
- Which events are the most frequent?
- What is the distribution of network traffic over time (hour of the day, day of the week, month of the year, etc.)?
- What is the user distribution over different domain areas?
- Are there and what are the differences in access for users from different geographic areas?

For further analysis, more questions need to be answered, such as:

- In what context are the components or features used?
- What are the typical event sequences?
- Are there any general behavior patterns across all users, and what are they?
- What are the differences in usage and behavior for different user population?
- Whether user behaviors change over time, and how?

Some of these questions can be answered by OLAP analysis. Others, however, will require more sophisticated data mining techniques, which will be examined in the following subsection.

3.3 Data mining on web log data cube and web log database

As discussed in Section 2, the WebLogMiner can perform various kinds of data mining tasks, including summarization, comparison, association, classification, prediction and time-series analysis. The first several kinds of data mining tasks have been discussed in our previous data mining research papers, such as [10, 11, 12]. Due to the limited space, the examples analyzed here are all related to the last data mining task: *time-series analysis*.

Time-series analysis is also the most important data mining task in the Web log analysis because all Web log records register time stamps, and most of the analyses are focused on time-related Web access behaviors.

Example 3.2 (Web traffic analysis) Web traffic analysis identifies the network load and traffic patterns in relevance to time. It helps us monitor system performance, overlook the trends of the amount of users activities, and determine the optimal server size. This analysis is performed by generalization of some attributes, such as traffic volume, along the time dimension. Drilling can be performed along the time dimension to obtain more detailed charts of the traffic fluctuation patterns along time.

Figure 4 shows the network traffic generated by a particular fieldsite over a four month period (January to April 1997). The chart in Figure 4 shows that the traffic had a general increasing trend over the four months, while within each month there are cycles of peaks and valleys indicating the traffic generally went up at the beginning of the week (Sunday) and during the week days (Monday to Thursday) and came down during the weekend (Friday and Saturday), as expected by our own experience. □

Example 3.3 (Typical event sequence and user behavior pattern analysis) Event sequence analysis lets us select the pattern across users and categorize users by their behavior patterns. This would help us understand how the system was used by different users, whether and where the users experienced difficulties in using the system, and how the users' behavior

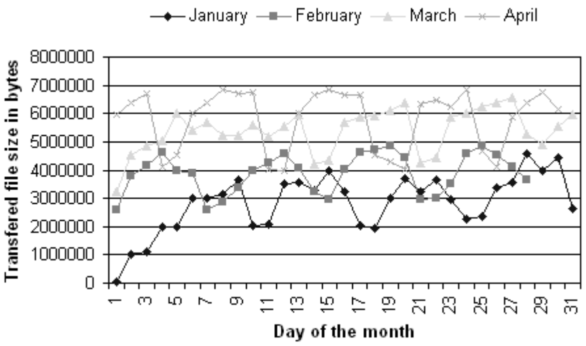

Figure 4: Network traffic generated by a particular fieldsite over a four month period

evolved over time. Event sequence can be plotted into timeline charts as outputs or being used for further analysis. From our analysis, we have formed different behavior patterns emerging from different groups. Figure 2 shows a typical behavior pattern for one use group. A set of events is listed on the left-hand side. The time intervals on the top are in the format of hour:minute:second. The bars running from left to right indicate the event sequence. The length of the bar indicates the duration of the the event.

One phenomenon we have discovered is that at the beginning of a course the users tended to explore many different system features. However, they become more and more focused over time.

While this phenomenon is intuitively understandable, it is interesting to find out what are the features that are consistently being used over time, what are the features that are being less used or not being used at all over time, and what are features that are being used more over time.

By incorporation of data cube techniques, all these features can be plotted into the time-related data flow charts and be examined in the context of multiple dimensions and multiple granularities. For example, usage patterns can be analyzed in the data cube not only based on one particular message view, but also based on repeated time framework. Drill-down and roll-up can be performed in such a framework to demonstrate regularities with different time attributes. □

Example 3.4 (Transition analysis) Transition analysis is similar to time-related association analysis, such as sequence analysis, studied in the literature [16]. Figure 5 shows a transition metrics for a given group of users over a certain period of time, which indicates the probabilities one event follows another. This probability is calculated based on the event frequency and event sequence. For example, the probability that one would re-start V-Groups after listing all messages is 0.2 for this group of users. This actually indicates a problem in the navigation which was not foreseen by V-Groups designers. Such metrics can be used to do further Markov Chain analysis which may give us insights of the access paths, starting and ending points, etc.

□

Example 3.5 (Trend analysis) Trend analysis helps us examine the patters between key events and tell us what the trend is between the two events. This allows us to detect whether there is any consistent pattern and whether an established pattern changes over time.

Figures 6 shows the event trees of a particular user's behavior from the first to the fourth month of using the system. All branches were started with one key event, "start V-Groups" and ended with the other key event, "display a message". The numbers in the parentheses are frequencies. The figures indicate that while the event patterns diverged frequently in the first month, they became much more converged over time. The figure indicates a strong event pattern of "start V-Groups", "list conferences", "list unread messages" and "display a message".

□

4 Discussion and Conclusions

The World-Wide Web is continuously growing and "collecting" all kinds of resources, text, multimedia, applications, etc. Despite the anarchy in which it is growing, the Web is one of the biggest repositories ever built. Analyzing the web access logs of different web sites can help understand the user behaviour and the Web structure, thereby improving the design of this colossal collection of resources. Therefore, it is important to build tools to analyze the web access patterns.

Currently available web log analysis tools report interesting statistics, but are limited by the huge sizes of the log files continuously expanding, the type of data collected in the web logs, and the techniques used to analyze this data. Consequently, it is imperative to design a good web log analysis tool that would overcome the current limitations of the web log and recommend directives for new web log standards that would help better analyze the web access trends and discover useful knowledge from the access records.

In this paper, we have outlined the design of our system WebLogMiner, which benefits from OLAP and data mining techniques, and multi-dimensional data cube, to interactively extract implicit knowledge from very large web log files. Concrete examples using these techniques were given for time-series pattern analysis. The major strengths of this design are its scalability, interactivity, and the variety and flexibility of the analyses possible to perform. Despite these strengths, the discovery potential of such a design is still limited due to the current impoverished web log files.

The web log file structure is still the same for most of the web servers as the structure of the first NCSA httpd server log. It is evident now that web servers should collect and enter more information in their logs. It is urgent to study specific needs in web access pattern analysis and recommend a new structure for web access logs. This new structure would significantly simplify the data cleaning and data transformation stage. Our experience showed us that the data cleaning and data transformation step is not only crucial, but also is the most time consuming. A good data filtering process needs metadata provided by web site designers

	display VU campus map	start V-Groups	List conf	list unread messages	load VU Welcome Page	display a message	initiate Add a message	list all messages	preview /add/cancel a message	Total
display VU campus map		1.000								1.000
start V-Groups			.988	.012						1.000
List conference				.823	.177					1.000
list unread messages		.150		.017	.017	.732	.042	.042		1.000
load VU Welcome Page	1.000									1.000
display a message		.304		.467	.163	.012			.054	1.000
initiate Add a message									1.000	1.000
list all messages		.200		.200	.200	.400				1.000
preview /add/cancel a message				.382	.048	.048			.524	1.000

Figure 5: A transition metrics

Figure 6: Event trees of months one to four.

to help interpret the access records and make effective assumption with regard to filtering irrelevant records and mapping the others to significant labels. We think there is a need for a web site metadata specification standard that would help designers to specify the structure of the resources delivered, and the web log analysis tools to interpret the web log entries. Such metadata specification standards could lead to the construction of generic tools for data cleaning and data transformation.

We are still in the implementation process of a WebLogMiner prototype. We plan to integrate the technology developed for our data mining system DBMiner[10] in WebLogMiner and use DBMiner's sparse data cube capabilities, and develop additional modules for time-series pattern analysis.

Due to the important size and the ever exploding nature of the web log files, the construction of the multi-dimensional data cube necessary for the on-line analytical processing and knowledge discovery, is very demanding and time consuming. To reduce the data cube construction time, we plan to build a bridge between the web server and a data cube that would propagate access information directly to the aggregated cells of the data cube. The incrementally updated web log data cube would allow real time web access analysis and data mining.

Further development and experiments with WebLogMiner will be reported in the future.

References

[1] S. Agarwal, R. Agrawal, P. M. Deshpande, A. Gupta, J. F. Naughton, R. Ramakrishnan, and S. Sarawagi. On the computation of multidimensional aggregates. In *Proc. 1996 Int. Conf. Very Large Data Bases*, pages 506–521, Bombay, India, Sept. 1996.

[2] D. Backman and J. Rubbin. Web log analysis: Finding a recipe for success. In *http://techweb.comp.com/nc/811/811cn2.html*, 1997.

[3] C. Brunk, J. Kelly, and R. Kohavi. MineSet: An integrated system for data mining. In *Proc. 3rd Int. Conf. Knowledge Discovery and Data Mining (KDD'97)*, pages 135–138, Newport Beach, California, August 1997.

[4] M. S. Chen, J. Han, and P. S. Yu. Data mining: An overview from a database perspective. *IEEE Trans. Knowledge and Data Engineering*, 8:866–883, 1996.

[5] M. S. Chen, J. S. Park, and P.S. Yu. Data mining for path traversial patterns in a web environment. In *Proc. 16th Int'l Conf. Distributed Computing Systems*, pages 385–392, May 1996.

[6] U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy. *Advances in Knowledge Discovery and Data Mining*. AAAI/MIT Press, 1996.

[7] R. Fuller and J. de Graaff. Measuring user motivation from server log files. In http://www.microsoft.com/usability/webconf/fuller/fuller.htm, 1997.

[8] W. Gong. Periodic pattern search in time-related data sets. In *M.Sc. Thesis, Simon Fraser University*, Burnaby, B.C., Canada, November 1997.

[9] J. Graham-Cumming. Hits and miss-es: A year watching the web. In *Proc. 6th Int. World Wide Web Conf.*, Santa Clara, California, April 1997.

[10] J. Han, J. Chiang, S. Chee, J. Chen, Q. Chen, S. Cheng, W. Gong, M. Kamber, G. Liu, K. Koperski, Y. Lu, N. Stefanovic, L. Winstone, B. Xia, O. R. Zaïane, S. Zhang, and H. Zhu. DBMiner: A system for data mining in relational databases and data warehouses. In *Proc. CASCON'97: Meeting of Minds*, pages 249–260, Toronto, Canada, November 1997.

[11] J. Han and Y. Fu. Discovery of multiple-level association rules from large databases. In *Proc. 1995 Int. Conf. Very Large Data Bases*, pages 420–431, Zurich, Switzerland, Sept. 1995.

[12] M. Kamber, J. Han, and J. Y. Chiang. Metarule-guided mining of multi-dimensional association rules using data cubes. In *Proc. 3rd Int. Conf. Knowledge Discovery and Data Mining (KDD'97)*, pages 207–210, Newport Beach, California, August 1997.

[13] R. Kimball. *The Data Warehouse Toolkit*. John Wiley & Sons, New York, 1996.

[14] M. Perkowitz and O. Etzioni. Adaptive sites: Automatically learning from user access patterns. In *Proc. 6th Int. World Wide Web Conf.*, Santa Clara, California, April 1997.

[15] J. Pitkow. In search of reliable usage data on the www. In *Proc. 6th Int. World Wide Web Conf.*, Santa Clara, California, April 1997.

[16] R. Srikant and R. Agrawal. Mining generalized association rules. In *Proc. 1995 Int. Conf. Very Large Data Bases*, pages 407–419, Zurich, Switzerland, Sept. 1995.

[17] T. Stabin and C. E. Glasson. First impression: 7 commercial log processing tools slice & dice logs your way. In *http://www.netscapeworld.com/netscapeworld/nw-08-1997/nw-08-loganalysis.html*, 1997.

[18] T. Sullivan. Reading reader reaction : A proposal for inferential analysis of web server log files. In *Proc. 3rd Conf. Human Factors & the Web*, Denver, Colorado, June 1997.

[19] L. Tauscher and S. Greenberg. How people revisit web pages: Empirical findings and implications for the design of history systems. *International Journal of Human Computer Studies, Special issue on World Wide Web Usability*, 47:97–138, 1997.

[20] Y. Zhao, P. M. Deshpande, and J. F. Naughton. An array-based algorithm for simultaneous multidimensional aggregates. In *Proc. 1997 ACM-SIGMOD Int. Conf. Management of Data*, pages 159–170, Tucson, Arizona, May 1997.