WebTool: An Integrated Framework for Data Mining

F. Masseglia[1,2], P. Poncelet[3,4], and R. Cicchetti[3,4]

[1] LIRMM, 161 Rue Ada, 34392 Montpellier Cedex 5, France
E-mail: massegli@lirmm.fr
[2] PRiSM - Univ. de Versailles, 45 Av. des Etats-Unis 78035 Versailles Cedex, France
[3] LIM - Faculté des Sciences de Luminy, , Case 901, 163 Av. de Luminy, 13288 Marseille Cedex 9, France, E-mail: {poncelet,ciccheti}@lim.univ-mrs.fr
[4] IUT Aix en Provence

Abstract. Large volumes of data such as user address or URL requested are gathered automatically by Web servers and collected in access log files. Analysis of server access data can provide significant and useful information for performance enhancement, and restructuring a Web site for increased effectiveness. In this paper, we propose an integrated system (WebTool) for mining user patterns and association rules from one or more Web servers and pay a particular attention to handling of time constraints. Once interesting patterns are discovered, we illustrate how they can be used to customize the server hypertext organization dynamically.

keywords: data mining, Web usage mining, sequential patterns, time constraints, association rules.

1 Introduction

With the growing popularity of the World Wide Web (Web), large volumes of data such as address of users or URLs requested are gathered automatically by Web servers and collected in access log files. Analysis of server access data can provide significant and useful information for performance enhancement, restructuring a Web site for increased effectiveness, and customer targeting in electronic commerce. Discovering relationships and global patterns that exist in access log files, but are hidden among the vast amounts of data is usually called Web Usage Mining [5, 15].

The groundwork of the approach presented in this paper addresses the problem of exhibiting behavioural patterns from one or more servers collecting data about their users. Our proposal pays particular attention to time constraint handling. We propose an integrated system for mining either association rules or sequential patterns. In our context, by analyzing informations from Web servers, an association rule could be, for instance, "*50 % of visitors who accessed URLs* `plaquette/info-f.html` *and* `labo/infos.html` *also visited* `situation.html`" or "*85% of visitors who accessed URLs* `iut/general.html` `departement/info.html` *and* `info/program.html` *also visited URL* `info/debouches.html`". Handling time constraints for mining sequential patterns could provide relationships such as: "*60 % of clients who visited* `/jdk1.1.6/docs/api/Package-java.io.html` *and* `/jdk1.1.6/docs/api/java.io.BufferedWriter.html` *in the same transaction, also accessed* `/jdk1.1.6/docs/relnotes/deprecatedlist.html` *during the following month*" or "*34 % of clients visited* `/relnotes/deprecatedlist.html` *between September the 20th and October the 30th*". Once interesting patterns are discovered, they can be used to dynamically customize the hypertext organization. More precisely, the user current behaviour can be compared to one or more sequential patterns and navigational hints can be added to the pages proved to be relevant for this category of users.

The rest of the paper is organized as follows. Our proposal is detailed in section 2. In section 3, we present a very brief overview of the implementation. Section 4 addresses the problem of using discovered patterns in order to customize the hypertext organization. Related work, presented in section 5, is mainly concerned with mining useful information from Web servers. Finally section 6 concludes with future directions.

2 Principles

For presenting our approach, we adopt the chronological viewpoint of data processing: from collected raw data to exhibited knowledge. Like in [5], we consider that the mechanism for discovering relationships and global patterns in Web servers is a 2-phase process. The starting point of the former phase is data automatically gathered by Web servers and collected in access log.

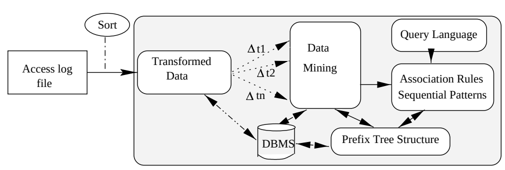

Fig. 1. An overview of the WebTool system

From such a file, the preprocessing phase removes irrelevant data and performs a clustering of entries driven by time considerations. It results in a populated database containing the meaningful remaining data. In the second phase, data mining techniques are applied in order to extract useful patterns or relationships and a visual query langage is provided in order to improve the mining process. Our approach is supported by an integrated system enforcing the described capabilities. Its architecture, close to that of the WebMiner system [5], is depicted in figure 1.

2.1 Data preprocessing

An input in the file log generally respects the *Common Log Format* specified by the CERN and the NCSA [4] and contains address IP of the customer, the user identifier, the access time, the method o f request (e.g. PUT/GET), the URL of the reached page, the protocol used, a possible error code and the number of transmitted bytes. Nevertheless, without loss of generality, we assume in the following that a log entry is merely reduced to the IP address which originates the request, the URL requested and a time stamp. Figure 2 illustrates a snapshot of the access log file from the Web server of the "IUT d'Aix en Provence".

```
132.208.12.150 - - [29/Nov/1998:18:02:30 +0200] "GET /info/RECRUT.gif HTTP/1.0"  200 1141
132.208.12.150 - - [29/Oct/1998:18:03:07 +0200] "GET /info/recrut.html HTTP/1.0"  200 1051
132.208.127.200 - - [16/Oct/1998:20:34:32 +0100] "GET /geaaix/home.html HTTP/1.0 " 200 14617
148.241.148.34 - - [31/Oct/1998:01:17:40 +0200] "GET /info/index.html HTTP/1.0"  304 -
148.241.148.34 - - [31/Oct/1998:01:17:42 +0200] "GET /info/recrut.html HTTP/1.0"  304 -
192.70.76.73 - - [22/Nov/1998:11:06:11 +0200] "GET /info/program.html HTTP/1.0"  200 4280
192.70.76.73 - - [22/Nov/1998:11:06:12 +0200] "GET /info/MATIERES.gif HTTP/1.0"  200 2002
192.93.19.14 - - [07/Dec/1998:11:44:15 +0200] "GET /queldept.html HTTP/1.0" 200 5003
```

Fig. 2. An example of entries in an access log file

During the data processing, three types of manipulations are carried out on the entries of the server log. First of all, a data filtering step is performed in order to filter out irrelevant requests. Then the remaining access log file is sorted by address and time. Finally, entries sufficiently close over time can be clustered. Most of Web log analysis tools operates a cleaning step during which they filter out requests for pages encompassing graphics as well as sound and video (for example, removing log entries with filename suffixes such as *.GIF, .JPEG*).

The WebTool system provides such cleaning facilities. Nevertheless, like in [15], we prefer to avoid their use because we believe that eliminated data may capture interesting and useful information about Web site structure, traffic performance, as well as user motivations. Of course, such a choice requires the implementation of efficient algorithms for extracting knowledge because the size of the access log file remains very large (during our experiments, we observe that removing pages encompassing graphics results in handled data size reduced from 40% to 85%).

The next step aims exhibiting users transactions and organizing data for an increased efficiency. It operates a sort of the file all along encoding data: URLs and visitors are mapped into integer, and date as well as time fields are expressed in relative time from the smallest date of the file.

In the market basket problem, each transaction is defined as a set of purchases bought by a customer at a time. In our context, user transaction has not counterpart because handled data does not capture user working session. Instead, each requested URL, in the access log file, is provided with a time stamp and could be seen as a single transaction. To avoid that situation, we propose like in [10] to cluster together entries, sufficiently close over time by using a maximum time gap (Δt) specified by user. Thus, the preprocessing phase results in a new database containing coded transactions. Each transaction provided with a relative data concerns a visitor, and groups together URLs visited during a common time range. Data can then be dealt for exhibiting knowledge.

2.2 Knowledge Discovery

This section widely resumes the formal description of the Web usage mining proposed in [10] and enhances the problem with useful information for handling time constraints proposed by [13]. From the transformed data yielded by the preprocessing stage, two techniques of knowledge discovery can be applied for fully meeting the analyst needs.

Mining association rules
The techniques used in mining association rules are generally applied in databases where each transaction is made up of a set of items. As we have already noticed in the preprocessing phase, it is necessary within the Web mining framework to gather the items between-them [10]. Let TA be a set of all association transactions obtained from Log. An association transaction t, $t \in TA$, is a tuple $t =< ip_t, UR_t >$ where UR_t, the URL set for t, is defined by $UR_t = ([l_1^t.url]...[l_m^t.url])$, such that for $1 \leq k \leq m$, $l_k^t \in Log$, $l_k^t.ip = ip_t$, $l_k^t.url$ must be unique in UR_t, and $l_{k-1}^t.url < l_k^t.url$.
In other words, an association transaction does not take into account transaction cutting and for each transaction, URLs are sorted in lexicographic order.

Definition 1 Let the database $D = \{t_1, t_2, ..., t_n\}$ be a set of n association transactions, each one consisting of a set of URLs, UR, and associated with a unique identifier corresponding to the visitor id ip_t. A transaction $t \in D$ is said to contain a set UR if $UR \subseteq t$. The *support* of UR is the percentage of transaction in D containing UR: $support(UR) = ||\{t \in D | UR \subseteq t\}|| / ||\{t \in D\}$. An association rule is a conditional implication among a set of URLs. The *confidence* of an association rule r: $UR_1 \Rightarrow UR_2$, where UR_1 is called the antecedent of the rule and UR_2 is called the consequent, is the conditional probability that a transaction contains UR_2, given that it contains UR_1. In other words, $confidence(r) = support(UR_1 \cup UR_2) \ / \ support(UR_1)$.

The problem of mining association rules in D is defined as follows. Given user defined minimum support and confidence, find all associations rules that hold with more than the given *minSupp* and *minConf*. This problem can be broken into two sub-problems [1]: (i) Find all frequent URs in D, i.e. URs with support greater or equal to minSupp. (ii) For each frequent set of URs found, generate all association rules with confidence greater or equal to minConf. The second sub-problem can be solved very quickly and in main memory in a straighforward manner once all frequent URs and their support are known. Hence, the problem of mining association rules is reduced to the problem of finding frequent URs and we focus, in the WebTool system, on how efficiently extract frequent URs.

Mining sequential patterns Taking into account time for mining sequential patterns requires defining the concept of sequence within the framework of the Web mining.

Definition 2 Let T be a set of all temporal transactions. A temporal transaction t, $t \in T$, is a triple $t =< ip_t, time_t, \{UT_1, UT_2, ..., UT_n\} >$ where for $1 \leq i \leq n$, UT_i is defined by $UT_i = ([l_1^t.url, l_1^t.time]...[l_m^t.url, l_m^t.time])$, such that for $1 \leq k \leq m$, $l_k^t \in Log$, $l_k^t.ip = ip_t$, $l_k^t.url$ must be unique in UT_t, $l_{k+1}^t.time - l_k^t.time \leq \Delta t$, $time_t = max_{1 \leq i \leq m} l_i^t.time$.

From temporal transactions, data sequences are defined as in [10]. Discovering sequential patterns resembles closely to mining association rules. However, elements of handled sequences are sets of URLs and not URL, and a main difference is introduced with time concerns. However, the above definition has the following limitations: the user often wants to specify maximum and/or minimum time gaps between adjacent URLs of the sequential patterns, or the user can decide that it does not matter if URLs were accessed separately as long as their occurrrences enfold within a given time window. Widely inspired from [13], a frequent sequence is defined as follows:

Definition 3 Given a user-specified minimum time gap (*minGap*), maximum time gap (*maxGap*) and a time window size (*windowSize*), a data-sequence $d =< UT_{t_1}^d \ UT_{t_2}^d \ ... \ UT_{t_m}^d >$ is said to *support* a sequence $S =< UT_{t_1}^S \ UT_{t_2}^S \ ... \ UT_{t_n}^S >$ if there exist integers $l_1 \leq u_1 < l_2 \leq u_2 < ... < l_n \leq u_n$ such that: (i) UT_i^s is contained in $\cup_{k=l_i}^{u_i} UT_k^d$, $1 \leq i \leq n$; (ii) $UT_{u_i}^d.time$ - $UT_{l_i}^d.time \leq windowSize$, $1 \leq i \leq n$; (iii) $UT_{l_i}^d.time$ - $UT_{u_{i-1}}^d.time > min\text{-}gap$, $2 \leq i \leq n$; (iv) $UT_{u_i}^d.time$ - $UT_{l_{i-1}}^d.time \leq max\text{-}gap$, $2 \leq i \leq n$. The *support* of s, $supp(s)$, is the fraction of all sub-sequences in D supporting s. When $supp(s) \geq minSupp$ holds, being given a *minimum support* value $minSupp$, the sequence s is called *frequent*.

Mining sequences with time constraints allows a more flexible handling of the visitor transactions, insofar the end user is provided with the following advantages: (i)To gather URL accesses when their dates are rather close via the windowSize constraint. For example, it does not matter if URLs in a sequential pattern were present in two different transactions, as long as the transaction-times of those transactions are within some small time window. The windowSize constraint is rather similar to that of Δt but it generally relates to a longer range of time (a few hours or a few days). (ii)To regard sets of URLs as too close or distant to appear in the same frequent sequence with the minGap or maxGap constraints. For example, the end user probably does not care if a visitor accesses URL "/java-tutorial/ui/animLoop.html", followed by "/relnotes/deprecatedlist.html" three months later.

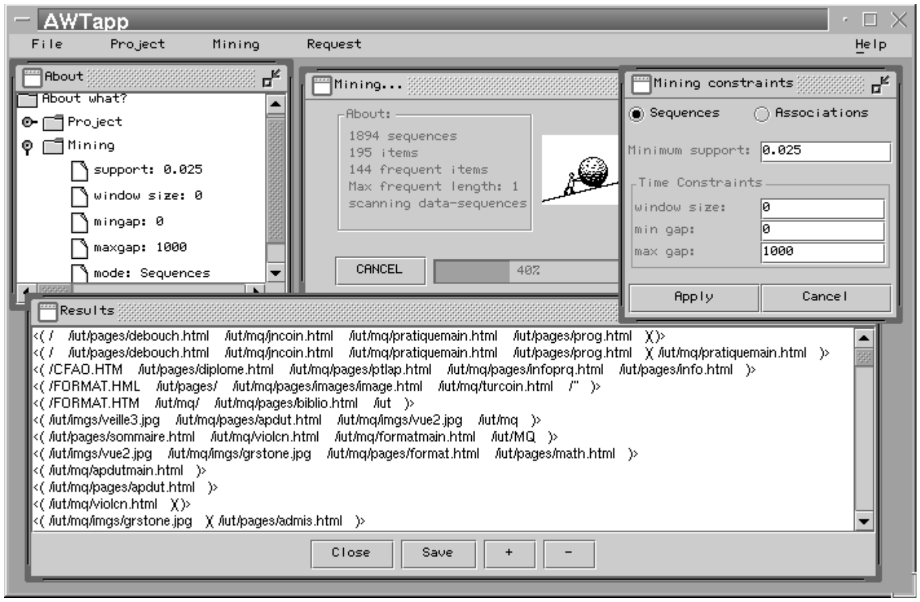

Fig. 3. A snapshot of the graphical interface of the WebTool system

An efficient algorithm for Web mining In the WebTool system we propose a very efficient algorithm which is described in [9]. The PSP algorithm (*Prefix-tree for Sequential Patterns*), used in the WebTool system was firstly defined for mining sequential patterns in market basket applications.

The principle fully resumes the fundamental principles of the GSP algorithm proposed in [13]. Its originality is to use a different hierarchical structure than in GSP in order to improve efficiency of retrievals of sequential patterns.

Arguing that the problem of the mining association rules is included in the mining sequential patterns problem, the principle adopted by WebTool is to use a common structure and the same algorithms to obtain association rules. The adaptation of PSP is done considering that all the transactions took place at the same time. Thus, during the application of PSP, transaction cutting is no longer considered and the yielded result is a frequent set of URLs. The rule generation from these sets of URLs is carried out by the visualization tool. In the figure 3, a snapshot of the WebTool system is depicted.

3 Experiments

We implemented the WebTool system on Ultra Sparc Station. Algorithms for mining association rules or sequential patterns are implemented using C++. The user interface module is implemented using Java (JDK 1.1.6). This module also concerns the preprocessing phase, i.e. the mapping from an access log file to a

database of data-sequences according to the user defined time window (Δt), and the visualization tool.
For instance, Let us consider the following association rule extracted by the mining process on the LIRMM access log file:`<(lirmm/plaquette/info-f.html lirmm-infos.html)>`$\Rightarrow$ `<(situ.html /autour.html mtp/index.html)>` *conf = 13*. It indicates that 13% of the visitor who obtained information about the laboratory LIRMM and more particularly about computer science, would like to know more about the geographical location of the laboratory (`situ.html`), how coming to LIRMM (`autour.html`) as well as informations on Montpellier (`mtp/index.html`).

4 Updating the hypertext organization dynamically

We developed a generator of dynamic links in Web pages using the rules generated from sequential patterns which is intended for recognizing a visitor according to his navigation through the pages of a server.
Since we are only interested in navigation through pages, we assume, in the following, that the hypertext document is defined as graphs with typed nodes and edges. An hypertext navigation for a visitor C is thus defined as a tuple $E_C =< id_C, \{n_1^{t_0}, n_2^{t_1}, ...n_{t_n}^{t_m}\} >$ where $1 \leq k \leq n$ and $1 \leq t \leq m$, n_k^t is the node accessed by the visitor and its associated time stamp, i.e. n_k is the URL of the reached page for the visitor C and the associated time.

Definition 4 Let us assume user defined parameters standing for the confidence (*conf*) and time constraints (Δt, *windowSize*, *minGap* and *maxGap*). A rule R is a triple $R =<< a_1\ a_2\ ...\ a_i >, < c_1\ c_2\ ....\ c_j >, conf_R >$ where $1 \leq k \leq i$, a_k stands for a set of URLs in the antecedent part, $1 \leq k \leq j$, c_k stands for a set of URLs in the consequent part and $conf_R$ is the confidence of R such as $conf_R \geq conf$ and the antecedent as well as the consequent part respect time constraints.

For performing the insertion of a dynamic link from the antecedent part of a rule, let us introduce the interesting subset notion.

Definition 5 Let us consider a rule R, and a user defined parameter $minPages$, standing for the minimal number of pages from which a link can be added. The *interesting subset* of R, noted Is_R, is defined as follows: $\forall a_k \in \{a_1\ a_2\ ...\ a_k\}$, $a_k \in Is_R$ if and only if $k \leq minPages$.

An hypertext navigation satisfying a rule is defined as follows:

Definition 6 Let us consider E_C the hypertext navigation of the client C. Let us consider a rule R. Let us consider the transformed paths of E_C according to time constraints, $E_{C_T} =< id_c, \{p_1,\ p_2,\ ...,\ p_l\} >$ where, for $1 \leq k \leq l$, p_k is a sequence encompassing sets of URLs grouped together according to Δt. Furthermore, $\forall p \in \{p_1,\ p_2,\ ...,\ p_l\}$, p respects time constraints. The client navigation E_C *satisfies* R if and only if $\exists p \in E_{C_T} \mid Is_R \subseteq_{seq} p$ where $\subseteq_{seq}$ stands for the inclusion of a sequence into another one [13].

Example 1 Let us consider the following visitor path: $p =< (X^{t0})\ (A^{t1}\ Y^{t2}\ B^{t3})\ (Z^{t4}\ C^{t5}) >$. Now, let us consider a rule R where the set of URLs of the antecedent part is: $a =< (A\ B)\ (C)\ (D\ E) >$. Let us assume that $minPages = 3$, thus to be considered as interesting three pages must be accessed by the same visitor. The *interesting subset*, IS_R, is the following $< (A\ B)\ (C) >$. The visitor satisfies the rule since $(A\ B) \subseteq (A^{t1}\ Y^{t2}\ B^{t3})$ and $(C) \subseteq (Z^{t4}\ C^{t5})$.

Implementation issues The technique presented so far was implemented using the functional architecture depicted in figure 4. The Web server (*http* daemon)

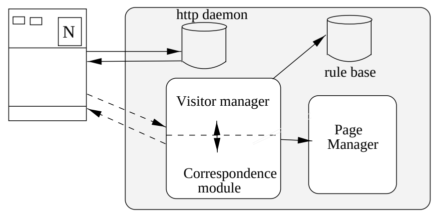

Fig. 4. General architecture

reacts to a customer request returning an applet encharged of the connection to the *visitor manager module* in order to transmit visitor IP address, required URL and a cookie encompassing the visitor navigation. The visitor manager module is a Java application running on the Web server site and using a client/server mechanism. When receiving IP address and required URL, the *visitor manager* examines the customer behaviour by using the *correspondence module*. The latter checks if the customer behaviour, i.e. the client navigation, satisfies a rule previously extracted by the data mining process. When an input satisfies a rule in the *correspondence module*, the required page is modified by the *page manager* which dynamically adds links towards the consequent of the recognized rule. The applet then recovers the URL and displays page on the navigator. If no rule corresponds to the current behaviour of the customer, the URL towards the required page is turned over to the applet which can display it.

Example 2 In the different rules obtained from the IUT access log file, we have noticed that 85% of visitors who visited the "Présentation générale de l'IUT" and the "Présentation générale du Département" pages in the same transaction, followed by the "Programme du Département Informatique" within 2 days, request the server on the "Débouchés avec un DUT" after an additional visit to the "Présentation générale du Département" (C.f. Figure 5). Let us consider a client accessing the pages <*(index.html info/genera.html) (info/program.html)*> during his navigation. Let us consider that the navigation satisfies the previous rule. A link corresponding to each consequent of this rule is added to the page. In our case, a link to the page "Débouchés" is dynamically inserted in the URL concerning the Program (C.f. Figure 5).

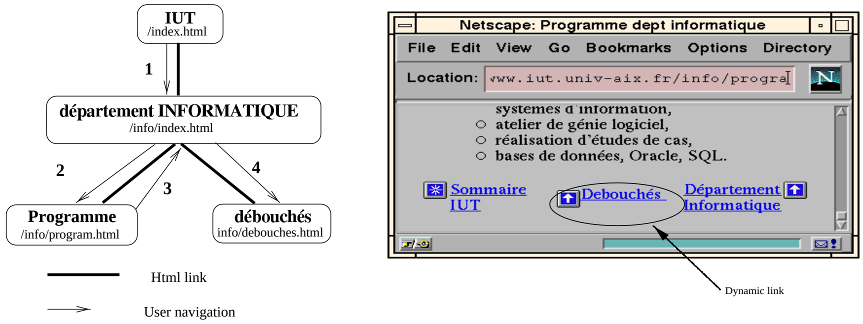

Fig. 5. Part of the hypertext organization and dynamically inserted link

5 Related Work

This section focuses on Web mining. The reader interested by an overview of data mining could refer to [1–3, 6, 11]. Using user access logs for exhibiting useful access patterns has been studied in some interesting approaches. Among them, we quote the approach presented in [10, 5]. A flexible architecture for Web mining, called WEBMINER, and several data mining functions (clustering, association, etc) are proposed. For instance, even if time constraints are not handled in the system (the minimum support is only provided), an approach for mining sequential patterns is addressed: an association rule-like algorithm [2], where the joining operation for candidate generation has been refined, is used. Various constraints can be specified using an SQL-like language with regular expression in order to provide much more control all along the discovery process. For example, the user can specify that he is only interested in clients from the domain **.edu** and in visits occurred after jan, 1, 1996. The WUM system proposed in [12] is based on an "aggregated materialized view of the Web log". Such a view contains aggregated data on sequences of pages requested by visitor. The query processor is incorporated to the miner in order to indentify navigation patterns satisfying properties (existence of cycles, repeated access, etc) specified by the expert. Incorporating the query language early in the mining process allows to construct only patterns having the desired characteristics while irrelevant patterns are removed. On-line analytical processing (OLAP) and multi-dimensional Web log data cube are proposed by [15]. In the WebLogMiner project, the data is split up into the following phases. In the first phase, the data is filtered to remove irrelevant information and it is transformed into a relational database in order to facilitate the following operation. In the second phase, a multi-dimensional array structure, called a data cube is built, each dimension representing a field with all possible values described by attributes. OLAP is used in the third phase in order to provide further insight of any target data set from different perspectives. In the last phase, data mining techniques can be used on the Web log data cube. The use of access patterns for automatically classifying users on a Web site is discussed in [14]. In this work, the authors identify clusters of users that access similar pages using user access logs entry. This lead to an improved organization of the hypertext documents. In this case, the organization can be customised on the fly and dynamically link hypertext pages for individual users.

6 Conclusion

In this paper, we presented an architectural framework for Web usage mining. We applied the approach for two differents servers and showed that association rules and sequential patterns extracted from Web server acces logs allows to predict user visit patterns and a dynamic hypertext organization. We are currently studying how to improve the process extraction using an incremental mining. This problem is very important in the Web mining context since the log files (access log, error log, etc) are always growing. We think that an incremental approach focusing on relationships previously extracted by a miner coud be very efficient.

References

1. R. Agrawal, T. Imielinski, and A. Swami. Mining Association Rules between Sets of Items in Large Databases. In *Proc. of the SIGMOD'93*, Washington, May 1993.
2. R. Agrawal and R. Srikant. Fast Algorithms for Mining Generalized Association Rules. In *Proc. of the VLDB'94*, Santiago, Chile, September 1994.
3. S. Brin, R. Motwani, and al. Dynamic Itemset Counting and Implication Rules for Market Basket Data. In *Proc. of the SIGMOD'97*, Tucson, Arizona, May 1997.
4. World Wide Web Consortium. In *http://lists.w3.org/Archives*, 1998.
5. R. Cooley, B. Mobasher, and J. Srivastava. Web Mining: Information and Pattern Discovery on the World Wide Web. In *Proc. of the ICTAI'97*, November 1997.
6. U.M. Fayad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors. *Advances in Knowledge Discovery and Data Mining.* AAAI Press, Menlo Park, 1996.
7. H. Mannila, H. Toivonen, and A.I. Verkamo. Discovery of Frequent Episodes in Event Sequences. *Data Mining and Knowledge Discovery*, 1(3), February 1997.
8. F. Masseglia. Le pré-calcul appliqué à l'extraction de sequential patterns en data mining. Technical report, LIRMM, France, June 1998.
9. F. Masseglia, F. Cathala, and P. Poncelet. The PSP Approach for Mining Sequential Patterns. In *Proc of the PKDD'98*, Nantes, France, September 1998.
10. B. Mobasher, N. Jain, E. Han, and J. Srivastava. Web Mining: Pattern Discovery from World Wide Web Transactions. Technical Report, Univ. of Minnesota, 1996.
11. A. Savasere, E. Omiecinski, and S. Navathe. An Efficient Algorithm for Mining Association Rules in Large Databases. In *Proc. of the VLDB'95*, Zurich, 1995.
12. M. Spiliopoulou and L.C. Faulstich. WUM: A Tool for Web Utilization Analysis. In *Proceedings of EDBT Workshop WebDB'98*, Valencia, Spain, March 1998.
13. R. Srikant and R. Agrawal. Mining Sequential Patterns: Generalizations and Performance Improvements. In *Proc. of the EDBT'96*, Avignon, September 1996.
14. T. Yan, M. Jacobsen, and al. From User Access Patterns to Dynamic Hypertext Linking. In *Proc. of the WWW Conference*, Paris, May 1996.
15. O. Zaïane, M .Xin, and J. Han. Discovering Web Access Patterns and Trends by Applying OLAP and Data Mining Technology on Web Logs. In *Proc. on Advances in Digital Libraries Conference (ADL'98)*, Santa Barbara, CA, April 1998.