Web Usage Mining: Contributions to Intersites Logs Preprocessing and Sequential Pattern Extraction with Low Support

Doru Tanasa

UNIVERSITÉ DE NICE SOPHIA ANTIPOLIS – UFR Sciences

Ecole Doctorale de Sciences et Technologies de l'Information et de
la Communication

THÈSE

pour obtenir le titre de

Docteur en Sciences
Spécialité Informatique

présentée et soutenue par

Doru Tanasa

Web Usage Mining: Contributions to Intersites Logs Preprocessing and Sequential Pattern Extraction with Low Support

Thèse dirigée par Brigitte TROUSSE
et préparée à l'INRIA Sophia Antipolis, projet AxIS

Rapporteurs: M. Henri BRIAND – M. Fabrice GUILLET
M. Patrick GALLINARI
M. Osmar ZAÏANE

soutenue le 3 juin 2005

Jury :

M.	Michel	RUEHER	Professeur	Président
M.	Henri	BRIAND	Professeur	Rapporteur
M.	Osmar	ZAÏANE	Professeur Associé	Rapporteur
Mme.	Maguelonne	TEISSEIRE	Maître de conférences	Examinateur
M.	Djamel	ZIGHED	Professeur	Examinateur
Mlle.	Brigitte	TROUSSE	Chargé de recherche	Directeur de thèse
M.	Florent	MASSEGLIA	Chargé de recherche	Invité

UNIVERSITÉ DE NICE SOPHIA ANTIPOLIS – UFR Sciences

Ecole Doctorale de Sciences et Technologies de l'Information et de la Communication

THÈSE

pour obtenir le titre de

DOCTEUR EN SCIENCES
Spécialité INFORMATIQUE

présentée et soutenue par

Doru TANASA

Fouille de données d'usage du Web : Contributions au prétraitement de logs Web Intersites et à l'extraction des motifs séquentiels avec un faible support

Thèse dirigée par Brigitte TROUSSE

et préparée à l'INRIA Sophia Antipolis, projet AxIS

Rapporteurs:	M. Henri BRIAND – M. Fabrice GUILLET
	M. Patrick GALLINARI
	M. Osmar ZAÏANE

soutenue le 3 juin 2005

Jury :

M.	Michel	RUEHER	Professeur	Président
M.	Henri	BRIAND	Professeur	Rapporteur
M.	Osmar	ZAÏANE	Professeur Associé	Rapporteur
Mme.	Maguelonne	TEISSEIRE	Maître de conférences	Examinateur
M.	Djamel	ZIGHED	Professeur	Examinateur
Mlle.	Brigitte	TROUSSE	Chargé de recherche	Directeur de thèse
M.	Florent	MASSEGLIA	Chargé de recherche	Invité

Abstract

The past fifteen years are characterized by an exponential growth of the Web both in the number of Web sites available and in the number of their users. This growth generated huge quantities of data related to the users interaction with the Web sites, recorded in Web log files. Moreover, the Web sites owners expressed the need to better understand their visitors in order to better serve them.

The Web Use Mining (WUM) is a rather recent research field and it corresponds to the process of knowledge discovery from databases (KDD) applied to the Web usage data. It comprises three main stages: the preprocessing of raw data, the discovery of schemas and the analysis (or interpretation) of results. A WUM process extracts behavioral patterns from the Web usage data and, if available, from the Web site information (structure and content) and on the Web site users (user profiles).

The quantity of the Web usage data to be analyzed and its low quality (in particular the absence of structure) are the principal problems in WUM. When applied to these data, the classic algorithms of data mining, generally, give disappointing results in terms of behaviors of the Web sites' users (e.g. obvious sequential patterns, stripped of interest).

In this thesis, we bring two significant contributions for a WUM process, both implemented in our toolbox, the AxisLogMiner. We propose a complete methodology for preprocessing the Web logs and a divisive general methodology with three approaches (as well as associated concrete methods) for the discovery of sequential patterns with a low support.

Our first contribution concerns the preprocessing of the Web usage data, which received less attention from the WUM research. The originality of the methodology for WUM preprocessing that we proposed consists in its *Intersites* aspect, essential to apprehend the behaviors of the users that navigate in a transparent way, for example, on several Web sites of the same organization. In addition to the integration of main existing work on this topic, we propose in our methodology four distinct steps: the data fusion, data cleaning, data structuration and data summarization. More precisely, we propose several heuristics for cleaning the Web robots, aggregated variables describing the sessions and the visits, as well as the recording of this data in a relational model. Several experiments were carried out, proving that our methodology allows a strong reduction (up to 10 times) of the initial number of requests and it offers richer logs, structured for the following stage of data mining.

Our second contribution aims at discovering from a large preprocessed log file the minority behaviors corresponding to the sequential patterns with low support. For that, we propose a general methodology aiming at dividing the preprocessed log file into a series of sub-logs. Based on this methodology, we designed three approaches for extracting sequential patterns with low support (the *Sequential*, *Iterative* and *Hierarchical* approaches). These approaches were implemented in hybrid concrete methods using algorithms of clustering and sequential pattern mining. Several experiments, carried out on logs collected from academic sites, enabled us to discover interesting sequential patterns having a very low support, while their discovery by a traditional algorithms was impossible.

Finally, we propose a toolbox the *AxisLogMiner*, which supports our preprocessing methodology and, currently, two of the hybrid methods for the discovery of sequential patterns in WUM. This toolbox was used to preprocess several log files and also to experiment on our methods implemented for extracting sequential patterns with low support.

Keywords: *Web usage mining (WUM), Web mining, data mining, Web access logs, WUM methodology, WUM preprocessing, intersites WUM, sequential pattern mining, low support, clustering, divisive methodology, WUM toolbox, Apriori-GST, AxisLogMiner*

Résumé

Les quinze dernières années ont été marquées par une croissance exponentielle du domaine du Web tant dans le nombre de sites Web disponibles que dans le nombre d'utilisateurs de ces sites. Cette croissance a généré de très grandes masses de données relatives aux traces d'usage du Web par les internautes, celles-ci enregistrées dans des fichiers logs Web. De plus, les propriétaires de ces sites ont exprimé le besoin de mieux comprendre leurs visiteurs afin de mieux répondre à leurs attentes.

Le Web Usage Mining (WUM), domaine de recherche assez récent, correspond justement au processus d'extraction des connaissances à partir des données (ECD) appliqué aux données d'usage sur le Web. Il comporte trois étapes principales : le prétraitement des données, la découverte des schémas et l'analyse (ou l'interprétation) des résultats. Un processus WUM extrait des patrons de comportement à partir des données d'usage et, éventuellement, à partir d'informations sur le site (structure et contenu) et sur les utilisateurs du site (profils).

La quantité des données d'usage à analyser ainsi que leur faible qualité (en particulier l'absence de structuration) sont les principaux problèmes en WUM. Les algorithmes classiques de fouille de données appliqués sur ces données donnent généralement des résultats décevants en termes de pratiques des internautes (par exemple des patrons séquentiels évidents, dénués d'intérêt).

Dans cette thèse, nous apportons deux contributions importantes pour un processus WUM, implémentées dans notre boîte à outils AxisLogMiner. Nous proposons une méthodologie générale de prétraitement des logs Web et une méthodologie générale divisive avec trois approches (ainsi que des méthodes concrètes associées) pour la découverte des motifs séquentiels ayant un faible support.

Notre première contribution concerne le prétraitement des données d'usage Web, domaine encore très peu abordé dans la littérature. L'originalité de la méthodologie de prétraitement proposée consiste dans le fait qu'elle prend en compte l'aspect multi-sites du WUM, indispensable pour appréhender les pratiques des internautes qui naviguent de façon transparente, par exemple, sur plusieurs sites Web d'une même organisation. Outre l'intégration des principaux travaux existants sur ce thème, nous proposons dans notre méthodologie quatre étapes distinctes : la fusion des fichiers logs, le nettoyage, la structuration et l'agrégation des données. En particulier, nous proposons plusieurs heuristiques pour le nettoyage des robots Web, des variables agrégées décrivant les sessions et les visites, ainsi que l'enregistrement de ces données dans un modèle relationnel. Plusieurs expérimentations ont été réalisées, montrant que notre méthodologie permet une forte réduction (jusqu'à 10 fois) du nombre des requêtes initiales et offre des logs structurés plus riches pour l'étape suivante de fouille de données.

Notre deuxième contribution vise la découverte à partir d'un fichier log prétraité de grande taille, des comportements minoritaires correspondant à des motifs séquentiels de très faible support. Pour cela, nous proposons une méthodologie générale visant à diviser le fichier log prétraité en sous-logs, se déclinant selon trois approches d'extraction de motifs séquentiels au support faible (*Séquentielle*, *Itérative* et *Hiérarchique*). Celles-ci ont été implémentées dans des méthodes concrètes hybrides mettant en jeu des algorithmes de classification et d'extraction de motifs séquentiels. Plusieurs expérimentations, réalisées sur des logs issus de sites académiques, nous ont permis de découvrir des motifs séquentiels intéressants ayant un support très faible, dont la découverte par un algorithme classique de type Apriori était impossible.

Enfin, nous proposons une boîte à outils appelée *AxisLogMiner*, qui supporte notre méthodologie de prétraitement et, actuellement, deux méthodes concrètes hybrides pour la découverte des motifs séquentiels en WUM. Cette boîte à outils a donné lieu à de nombreux prétraitements de fichiers logs et aussi à des expérimentations avec nos méthodes implémentées.

Mots Clefs : *Web usage mining (WUM), journaux d'accès Web, méthodologie WUM, prétraitement WUM, WUM multi-sites, fouille de données Web, fouille de données, extraction des motifs séquentiels, support faible, classification non-supervisée, méthodologie divisive, boîte à outils WUM, Apriori-GST, AxisLogMiner*

‘Would you tell me, please, which way I ought to go from here?’
‘That depends a good deal on where you want to get to,’ said the Cat.
‘I don’t much care where –’ said Alice.
‘Then it doesn’t matter which way you go,’ said the Cat.
‘– so long as I get somewhere,’ Alice added as an explanation.
‘Oh, you’re sure to do that,’ said the Cat, ‘if you only walk long enough.’

Lewis Carrol, *Alice’s Adventures in Wonderland*

”– Voudriez-vous me dire, s’il vous plaît, par où je dois m’en aller d’ici ?’
– Cela dépend beaucoup de l’endroit où tu veux aller.
– Peu m’importe l’endroit...
– En ce cas, peu importe la route que tu prendras.
– ... pourvu que j’arrive quelque part, ajouta Alice en guise d’explication.
– Oh, tu ne manqueras pas d’arriver quelque part, si tu marches assez longtemps.”

Lewis Carrol, *Alice au pays des merveilles*

Remerciements

Je tiens à remercier Brigitte Trousse qui a encadré mon travail de thèse dans l'équipe AxIS à l'INRIA Sophia Antipolis. Sur le plan professionnel, sa vaste expérience et ses précieux conseils m'ont aidé à bien mener ce travail à son terme. Sur le plan humain, avec sa disponibilité, sa patience et sa bonne humeur, elle a su créer une très bonne ambiance dans l'équipe qui a servi de cadre à mes recherches.

Je remercie également Florent Masseglia qui a participé à l'encadrement de cette thèse en me faisant ainsi partager sa grande expérience dans le domaine d'extraction de motifs séquentiels. Plus particulièrement, je le remercie pour son apport à la méthode et au logiciel "Divide & Discover".

Je suis très sensible à l'honneur que m'a fait le Professeur Michel Rueher en acceptant d'être le président de ce jury de thèse.

J'exprime toute ma gratitude aux professeurs Osmar Zaïane, Henri Briand, Fabrice Guillet et Patrick Gallinari pour avoir accepté de juger ce travail. Plus particulièrement je tiens à remercier le Professeur Osmar Zaïane pour ses précieux commentaires et les discussions que nous avons eues pendant sa visite dans l'équipe AxIS.

Mes sincères remerciements vont aussi à Madame Maguelonne Teisseire et au Professeur Djamel Zighed pour l'honneur qu'ils m'ont fait en acceptant de participer au jury de soutenance.

Un grand merci à tous mes anciens et actuels collègues d'équipe et plus particulièrement à Mihai pour son aide sur le développement des logiciels, et à Sergiu pour nos discussions sur le prétraitement des fichiers logs.

Je ne peux terminer ces remerciements sans mentionner mes proches auxquels je dédie cette thèse et dont l'amour et le soutien m'ont aidé à finir ce travail.

Enfin, merci à toi, Alina, pour tes relectures assidues et ton soutien tout au long de cette thèse.

Contents

List of Figures

List of Tables

List of Abbreviations

Some of the abbreviations and acronyms used throughout this thesis are listed below:

AR	Association Rule
BS	Browsing Speed
C&D	Cluster and Discover
CLF	Common Log Format
CSP	Contiguous Sequential Patterns
DB	Database
DCM	Dynamic Clustering Method
D&D	Divide and Discover
DS	Data Structure
ECLF	Extended Common Log Format
GST	Generalized Suffix Tree
H&D	Hierarchical Discover
HTTP	Hypertext Transfer Protocol
INRIA	The French National Institute for Research in Computer Science and Control
IP	Internet Protocol, also used as IP address
KDD	Knowledge Discovery in Databases
MCA	Multiple Component Analysis
MF	Maximal Forward (algorithm)
OAT	Online Adaptive Traversal
PCA	Principal Component Analysis
SL	Sub-log
SP	Sequential Pattern
SPM	Sequential Pattern Mining
URI	Uniform Resource Identifier
W3C	World Wide Web Corporation
WR	Web Robot
WUM	Web Usage Mining
WWW	World Wide Web
XGMML	eXtensible Graph Markup and Modelling Language

Chapter 1

Introduction

The World Wide Web (WWW) had an impressive development in the past fifteen years, today reaching more than 8 billion documents online [Goo05] and having 20 million new Web pages published each day [Tur02]. This phenomenal growth of the Web triggered the development of new domains of application, the Web Mining being one of them. The Web Mining is the process of discovering potential useful and previously unknown information from the Web data [KB00]. When this Web data consist of Web documents and resources, the process is referred as the *Web Content Mining* [CSM97].

Moreover, with more than 60 million Web sites and 800 million Internet users [Net05], the Internet activity resulting from their interaction generates a huge quantity of data recorded in Web access log files. The owners of the Web sites, especially commercial Web sites, are interested in obtaining more information about their customers in order to better target the cross-marketing campaigns, to display the relevant ads to their users and to reorganize the Web site for a smoother navigation [CSM97]. *Web Usage Mining (WUM)* applies data mining procedures to analyze the user access to Web sites. The WUM term was introduced by Cooley et al. in 1997 [CSM97]. In [MTP01b], the term of *"Intersites WUM"* was introduced for reffering to a WUM process where Web access logs from partner Web sites are incrementally analyzed. In [TT04a], we used this term for referring to a particular case of Web Usage Mining, when the WUM process applies to a Web site composed of several Web servers (i.e. several Web access logs).

Intersites WUM deals with the Web server logs gathered from several Web sites, generally belonging to the same organization. Today, an important organization might have several Web servers for its Web sites. For example, INRIA has one main Web server (`www.inria.fr`), then one server for each of its research units in France (see `www.inria.fr/inria/unites.en.html` for the complete list), and other servers for the Web sites' search engines or the intranet. A user navigates through all these servers transparently because the pages from different Web servers are strongly interlinked. Visitors might not even notice that the Web server has changed (to see this, they must look at the browser's address bar).

However, for WUM analysts performing a user-centric analyze, this change is important because they want to have a complete picture of the users' behaviors on the entire Web site. Thus, analysts must reassemble the users' paths through all the different Web servers they visited.

In this chapter, after enouncing and defining the main terms used in this research and in the WUM field, we discuss briefly the WUM process and each of its three steps. Then, we present the main applications of WUM as well as the current issues arising from this field. Finally, we enounce the main contributions of our thesis and the thesis structure.

1.1 Definitions

In accordance with the World Wide Web Consortium's (W3C) work on Web characterization terminology [LN99], we revised some of their definitions from the WUM perspective. We also propose new definitions of other three terms related to the WUM process (visit, episode, and Web server log file).

A *resource*, according to the W3C's Uniform Resource Identifier (URI) specification, can be "anything that has identity" [BLFM98]. Moreover, an URI is "a compact string of characters for identifying an abstract or physical resource" [BLFM98]. Possible examples include an HTML file, an image, and a Web service.

A *Web resource* is a resource accessible through any version of the HTTP protocol (for example, HTTP 1.1 or HTTP-NG).

A *Web server* is the server that provides access to the Web resources.

A *Web page* is the set of data constituting one or several Web resources that can be identified by an URI. If the Web page consists of n resources, the first $n-1$ are embedded into the n^{th} URI, which identifies the Web page.

A *page view* (also called *hit*) occurs at a specific moment in time, when a Web browser displays a Web page.

A *Web browser* or *Web client* is a client software that can send Web requests, handle the responses, and display requested URIs.

A *user* is a person using a Web browser.

A *Web request* is a request that a Web client makes for a Web resource. It can be *explicit* (user initiated) or *implicit* (Web client initiated). Explicit Web requests (also called *clicks*) are classified as *embedded* (i.e. the user selected a link from a Web page) or *user-input* (i.e. the user manually initiates the request – for example, by typing the address in the address bar or selecting the address from the bookmarks or history). Implicit Web requests are generated by the Web client that needs the embedded resources from a Web page (images, multimedia files, script files, etc.) in order to display that page.

A *user session* consists in a delimited number of a user's explicit Web requests across one or more Web servers.

A *visit* represents a subset of consecutive page views from a user session occurring *close enough* (measured by means of a time threshold or a semantical distance between pages).

An *episode* is a subset of a visit constituted from related clicks. For example, a user's visit to yahoo.com could consist of three distinct episodes: ordering a sport item, checking stock values, and searching for pictures of the latest Ferrari F60.

```
192.168.0.1 - - [29/Sep/2004:17:10:31 +0200] "GET /axis/people.shtml HTTP/1.1"
200 8289 "http://www-sop.inria.fr/axis/table.html" "Mozilla/4.0 (compatible; MSIE
6.0; Windows NT 5.2; .NET CLR 1.1.4322)"
```

Figure 1.1: A Web Request from INRIA's Web Server Log (in the ECLF Format)

A *Web server log file* contains requests made to the Web server, recorded in a chronological order. The most popular log file formats (developed by the CERN and the NCSA) are the *Common Log Format* [W3C95] (CLF) and an extended version of the CLF, *Combined Log Format*, noted, for distinction, ECLF and described in [NCS95]. A line in the ECLF (Figure 1.1) contains:

- The client's host name or its IP address,
- The client inetd id (generally empty and represented by a "-"),
- The user login (if applicable),
- The date and time of the request,
- The operation type (GET, POST, HEAD, etc.),
- The requested resource name,
- The request status,
- The requested page size,
- The user agent (a string identifying the browser and the operating system used), and
- The referrer of the request which is the URL of the Web page containing the link that the user followed to get to the current page.

The Web access log line from Figure 1.1 shows that a user from the IP address 192.168.0.1, successfully requested the page `/axis/people.shtml` on September 29th, 2004 at 17:01 PM. The user arrived on this page by selecting a link from the Web page `http://www-sop.inria.fr/axis/table.html` and used Microsoft Internet Explorer 6.0 to display the page.

From the terms that we introduced in this section, the ones referring to the usage (user, session, visit, episode), were previously defined and employed with different meanings by the authors in the WUM literature and the W3C's Web Characterization Authority (W3C's WCA) [LN99]. We summarized the differences between their definitions and ours in the Table 1.1.

Term	**W3C's WCA**	**WUM Literature**	**Our Terminology**
User	Person using a browser	Login or cookie or IP or (IP, User Agent)	Login or IP
User session	Delimited user requests over multiple servers	Delimited user requests on one server	(User, IP, User Agent) over analyzed servers
Visit	Server session	–	Time delimited requests in an user session
Episode	Related user requests	Related user requests	Conceptually related user requests in a visit

Table 1.1: Terminology Comparison Table

1.2 General Process of WUM

Web servers collect large volumes of data from the Web sites' usage. This data is stored in Web access log files. Together with the Web access log files, other data can be used in Web Usage Mining like the Web structure information, user profiles, Web page content, etc.

As [Coo00], in Figure 1.2, we divide the WUM in three main steps: preprocessing, pattern discovery and pattern analysis. The Web site structure information could be used in the preprocessing task, for example to generalize Web pages (i.e. replace multiple pages with a higher level index page). Moreover, when analyzing the patterns discovered, the site structure could be used to highlight "unexpected" patterns, i.e. patterns having high link distances between their Web pages.

The *link distance* between two pages A and B is the minimum number of hyperlinks the user has to traverse to go from A to B. The *depth* of a Web page A is defined as the link distance between the Web site's homepage and A.
In the following sections, we describe the three steps of the WUM process.

1.2.1 Preprocessing

The preprocessing task within the WUM process involves cleaning and structuring data to prepare it for the pattern discovery task. Web usage data is subject to a lot of noise

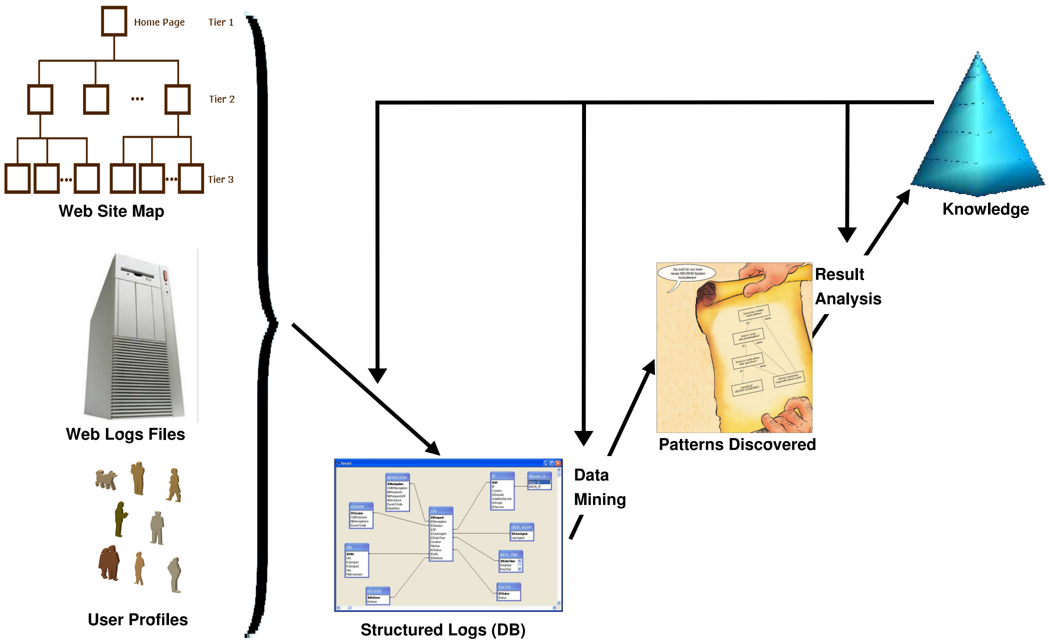


Figure 1.2: Schema of a General WUM Process

and missing data. The Web site's structure, its content and the Web server technologies behind the Web site heavily influence on the nature and size of the usage data collected. A "classical" Web site delivering information mainly through plain HTML pages will generate less usage data than a Web site with a lot of graphic content (considering the same number of visitors and hits).

1.2.1.1 Data Cleaning

The first preprocessing task is the data cleaning. Here, all irrelevant and noisy data are eliminated from the log files. Usually, requests for images and multimedia files are filtered out as well as Web robots' sessions. When requesting a Web page containing additional Web resources like images or script files, several implicit requests will be generated by the Web browser. If these requests are still present when the data mining step is performed, uninteresting patterns like *"Page, Image1, Image3, Image6"* may be found, making the pattern analysis step more complex. Also, Web robots, generally, have a predefined (programmed) behavior and the analysts are not interested in mining these requests.

1.2.1.2 Data Structuration

During this step, the requests from the raw log file are grouped by user, user session, page view, visit and episode.

Grouping requests by user, also called *user identification*, depends on the Web site policies. For Web sites requesting registration, the user identification task is straightforward and guaranteed to be correct. For all other methods (e.g. using the IP address or cookies), the accuracy of the user identification is not guaranteed.

The *user session* contains all the requests of a user, made from the same computer and having the same user agent.

The *page view identification* is important as it allows the analyst to select the explicit requests from the Web access log file. As defined in Section 1.1, the page view represents the answer of the Web browser to the user action (click). Usually, the Web browser generates several implicit requests for each explicit Web request from the user. The reason behind this is that the HTTP protocol allows only one file per request, while Web pages are generally composed of additional resources as images, multimedia files or other HTML files (in frames). At the end of the page view identification process only one request per page view has to be kept (i.e. the explicit one).

Visits and *episodes identification* depend on the purpose of the analysis. For some analysis, grouping the requests in user sessions and keeping one request per page view is enough and there is no need to further split the user session. But, depending on the period analyzed, one user session may span over several months and, therefore, the requests will not be related. In this case, we need to split the user session after a certain time lapse. Furthermore, in Web portals like Yahoo, one user may view during a short period, several Web pages that have completely different purposes (e.g. e-mail, online shopping, stock values). These pages will all be grouped under a single visit and analyzed together although they are not related. Therefore, it is better to group these pages in episodes according to their content and analyze them separately.

At the end of the data structuration step, the raw log file is transformed in one or several structured file(s) (or in a relational database) and only the necessary requests (for a WUM analysis) are kept from the original raw file. This can significantly reduce the size of the initial log file (e.g. down to 10-25% of the initial size).

1.2.2 Pattern Discovery

Once the raw logs have been preprocessed, data mining techniques can be applied on the dataset to discover new patterns. Such techniques include, but are not limited to: association rules mining, sequential pattern mining and clustering.

1.2.2.1 Association Rule Mining

The algorithms for mining association rules (ARs) were first developed for the market-basket analysis. Apriori [AS94] is the first and still the most used algorithm for this task. By using this algorithm, we can extract interesting correlations from the data, like a list of items that are frequently bought together.

Applying ARs to Web usage data means extracting items (i.e. Web pages) that frequently occur together. In this case, the notion of *frequent pattern* depends on a minimum *support* expressing the minimum number of *transactions* (i.e. user sessions, visits or episodes) that needs to be contained in the items forming the AR. Thus, association rules express frequent co-occurrences of Web pages together.

The general form of an AR is: $X => Y$, where X and Y are sets of Web pages. A support of $s\%$ for this AR means that X and Y are contained together in $s\%$ of the transactions. An AR has also a *confidence* associated with it. The confidence is calculated as the ratio between the support of the $X => Y$ rule and the number of transactions containing X. This parameter expresses the probability that the set of pages Y is visited when the set X is visited. For example, consider that the following association rulea has a confidence of 50%:

"When users visit the Job opportunities Web page, they will also visit the research teams index page".

This means that the numbers of transactions containing the "Job opportunities" page is twice superior to the number of transactions containing both pages.

Association rules represent a valuable result as they allow to make useful recommendations to Web sites' users depending on the pages they visited.

The main drawback of the ARs is that the order of the Web pages in the sessions is not considered and a session where the page A is visited before the page B is considered in the same way as a second session where the page B was visited before the page A. But the fact that the users visited page A before going to page B is essential for predicting the next request or for finding users with similar browsing behaviors. The sequential patterns represent one of the solutions to this problem.

1.2.2.2 Sequential Pattern Mining

Mining sequential patterns (SPs) is highly similar with mining association rules. In fact, the research in this area was also initiated by Agrawal & Srikant [AS95]. The main difference between the SPs and the ARs is that now the time element is taken into account (order of events, i.e. clicks). With sequential pattern extraction, the analyst may discover that:

"10% of the site users visited page A, then page B and then page C".

In this case, the support for the above SP is constituted only from the sessions containing the pages A, B, C following this precise order. Generally, the measure of *confidence* is not used for a sequential pattern, although it can be calculated by considering the SP as a special case of an AR with the first $n-1$ items being the set X and the last item being the set Y, in an AR with the general form $X => Y$.

Because of the order constraint required for a sequential pattern, the support is generally lower than the support of a corresponding association rule containing the same pages. Depending on the number of different Web pages in the Web site and the number of the Web site visitors, we need to mine the SPs with support values as low as 0.1%. In Chapter 3, we discuss in detail these issues of very low support when mining SPs from Web usage data.

1.2.2.3 Clustering

Clustering represents the division of data into groups of similar objects [Gor99]. Standard clustering techniques can be applied for grouping *similar* user sessions or Web pages. The notion of similarity is expressed using a *distance measure* usually defined on the set (or sequence) of the sessions' page views.

To prepare the Web usage data for clustering, each session is transformed into vectors of n elements, where n corresponds either to the number of pages in the Web site or to the number of page views from that session. Various types of distance measures may be used to calculate the similarity between the sessions encoded in this way. But computing the distance measures for the Web usage data proved to be rather complicated because of the high number of Web pages and sessions for a Web site. Moreover, expressing similarity between heterogeneous Web pages is also a complex problem.

However, clustering Web usage data allows the Web master to identify groups of users with similar behaviors for which personalized versions of the Web site may be created.

A survey of the main clustering techniques used in WUM is presented in Chapter 3, in Section 3.8.2.

1.2.3 Pattern Analysis

The pattern analysis is the final step of the WUM process. As in any other KDD process, the WUM has an unknown result and the analysis of its result allows the analyst to distinguish interesting results from non-interesting ones. The issue with WUM results is that it is extremely hard to capture and define the notion of *interestingness*. This vary according to the analysts beliefs, their needs, the Web site type and its structure, the user sessions analyzed, etc.

In this step of the WUM process, the analyst is interested in projecting the patterns discovered on the Web site structure or on its content. For instance, the analyst may be interested to see:

- On which sections of the Web site are situated the pages contained in the most frequent sequential patterns or

- What is the overlap between the clusters of the Web pages and their actual classification in the Web site hierarchy or

- What is the link distance between the first page and the last page of a sequential pattern.

Therefore, a visual tool for exploring the results helps the analyst in making the decision of keeping or dropping one result depending on its relevance. The set of patterns validated are then used for improving the Web site or for making online recommendations to new Web site visitors as described in the following section.

1.3 WUM Applications

The increasing interest in the Web Usage Mining domain, from both research and business communities, is explained by the high number of its possible applications. Although an interesting topic, the application of WUM results goes beyond the subject of our thesis. We enumerate some of the most important applications, but the interested reader may refer to [FL05, SCDT00] for detailed surveys of the domain:

- *Web Personalization* groups the techniques that deliver personalized Web content (pages) depending on the user profile (or previously visited pages). It can have different forms, such as a recommender system [JT98, MCS00] or an intelligent/adaptative Web site [VBYA04, BAC04]. A *recommender system* will propose links of possible interest to the user. For instance, if a visitor requests a paper about Web Usage Mining on INRIA's Web site, the system may suggest him/her to visit the home page of the AxIS research team[1]. The *intelligent (adaptative) Web site* is based on an idea proposed by Perkowitz and Etzioni [PE98]: a Web site may adapt its content to each user visiting the Web site in order to deliver personalized information.

- *Site improvement* that may be achieved either by modifying the *logical structure* (by adding new links) or the *physical structure* (modifying the organization of the Web pages in sections) of the Web site. The modification will be done depending on the access patterns of the Web site users.

- *Intelligent Web caching* represents another possibility for improving the quality of service for a Web site as the pages are delivered faster to the visitor when using this kind of system. It is well known that users are less likely to spend time on a slow Web site. By using the results of a WUM analysis, an Intelligent Web caching system is capable of predicting the next request for a user and, therefore, to anticipate and load in a cache (or buffer) the Web page to be requested [BGG$^+$01]. In this way, the user will not have to wait for the page to be loaded from the disk and the speed of the Web site is improved.

The number of WUM applications is growing continuously, especially due to the business interest in e-commerce Web sites and the related Web-marketing applications. Moreover, the growing interest in the Web semantic field and the recent field of Web semantic mining will bring new perspectives for the WUM-related applications.

[1] AxIS research team from INRIA Sophia Antipolis (`http://www-sop.inria.fr/axis/`) focuses on research in the WUM field.

1.4 Open Problems in WUM

During the last 15 years, storing and transmitting data has become cheaper. As a result, nowadays, databases containing terabytes (TB) of data are common in many domains such as astronomy, geography, biology, e-commerce, etc. For instance, in the e-commerce field, the Amazon.com Web site collects more than 10 TB of data daily [Wei04]. This data represents the information about the new users (personal details), orders (list of items ordered), user sessions (sequences of page views), and page views' contents (all the dynamic pages displayed). As this data comes from different sources and in different formats (content of Web pages, user profiles, sequences of Web pages), a structure and an integration process is necessary before conducting any WUM analysis. The issues related to integrating and analyzing this heterogenous and complex data are similar to those studied in the Mining Complex Data field [ZSD03, TMGB05].

For Amazon.com, the high increase in the amount of data collected is due to the growing number of users/customers browsing its Web site and to the growing number of products being offered (each new product generating at least one new Web page). But, this trend of increasing data quantity was not followed by a similar one in the data processing capabilities. For instance, extracting sequential patterns from such a volume of data (like in the Amazon.com example) is still a time consuming process or even an impossible process in some cases.

The Web Usage Mining is a recent research field of Data Mining. The fast and dynamic development of WUM left little time to the Data Mining analysts to understand and solve the problems arising in this field. Research dealing with the Web usage data must overcome a number of issues. We list below the current main WUM open problems:

1. The quantity of data is continuously increasing;
2. The preprocessing step does not receive enough analysis efforts;
3. The Web sites have no or little semantic definitions for their Web pages;
4. The sequential pattern mining techniques for WUM are not appropriate for dealing with the specifics of Web usage data, mainly with its huge quantity;
5. The SPM techniques often provide short and uninteresting results;
6. The three steps of the WUM process are not coordinated to create a coherent and unique process.

As previously mentioned, the main problem for the WUM analysis is represented by the **huge quantity of data currently collected**. More and larger Web sites and more visitors are the main causes for this. Moreover, Web sites' owners are struggling to keep and increase the number of their visitors as this is directly related to the profits the Web sites generate. Thus, **understanding the needs of their users** is vital for the Web sites' owners.

Rushing to analyze usage data without a proper preprocessing method will lead to poor results or even to failure. This was the case for some of the first WUM tools that were designed to directly extract association rules or sequential patterns from the list of Web resources logged for one IP address. Without properly cleaning, transforming and structuring the data prior to the analysis, one cannot expect to find meaningful knowledge. According to a `kdnuggets.com` survey, in a KDD process, the preprocessing step represents at least 60% of the entire process for about two thirds of the Data Mining experts responding to the survey[2].

Very often, the WUM methods just "translate" the logs into the proper input format for the DM algorithm. However, this would negatively affect the entire process. As an example, let us consider the following hypothetic situation: a Web access log with two requests one for `http://website/A/page.htm` and another one for `http://website/A/B/C/../../page.htm`. The two requests are in fact for the same resource ("../" is interpreted as the parent folder), but they would be treated as two distinct URLs, by most of the translation tools. Therefore, the first problem we aimed to solve was the **lack of a complete methodology for preprocessing in WUM**.

During our experiments with different data sets from various Web sites, we noticed one common problem for all the Web sites: the **Web sites are less structured** compared to the more classical Information Systems. For instance, Web pages discussing a similar topic are often placed in different locations on the Web site hierarchy or more topics are covered in the same page. The semantic topics (see Section 2.5.5) represent an efficient solution for page and Web site description. Through their hierarchy, they allow to structure the Web site pages. A semantic tag associated to each page would allow the WUM analyst on one hand to easily extract episodes from visits and, on the other hand, to automatically highlight interesting results in the result analysis phase. However, most of the Web sites have heterogenous pages without a common strategy for Web page development. Although automated tools for extracting Web sites hierarchy exist [MS01, MBGMF04], they represent only a partial solution.

Concerning the sequential pattern mining for WUM, **the SPM techniques available today are limited with respect to the quantity of data they can mine or to the size of the results** while the sizes of the Web access logs are apparently not, as previously seen. Adopted solutions, such as limiting the amount of data analyzed at once (for example considering only one day per month), or reducing the complexity of the data by using a Web page generalization are not an answer to the main WUM question: *"What do ALL my data say?"*. In order to answer this question, we need to design new methodologies for sequential pattern mining that are able to **deal with huge quantities of data**.

Moreover, if we consider a classical algorithm for the sequential pattern extraction, applied to logs from a large Web site, usually we find **short and uninteresting patterns** like *"10% of users visited the home page and then the search page"*. Obviously,

[2] http://www.kdnuggets.com/polls/2003/data_preparation.htm

this is an unsurprising and not a very interesting result as one would expect. The main reason for obtaining such results is that, for a large Web site, there is only a small number of people with similar behaviors and thus only these kind of frequent patterns (short and involving pages with a high number of requests) are the most common ones among the majority of the users. In order to **find more interesting patterns**, we need to be able to extract even those patterns corresponding to a small group of visitors, i.e. having a very low support.

Finally, another important aspect of the WUM process is the **coherence that should exist between its three steps**: preprocessing, data mining and result analysis. In our opinion, there is a need for more research effort to solve this issue. The three steps should be considered as being part of one single process and, therefore, when defining the preprocessing options, the data mining and the result analysis should also be taken into account. An integrated WUM framework will allow the users to redesign the whole WUM process based on the results obtained from a first process run, by modifying the preprocessing options and then re-running the DM algorithms. For instance, after running a first WUM analysis, the analysts may obtain patterns that refer mostly to a particular section of the Web site. Based on these results, they can decide to select in the preprocessing step only the pages corresponding to that section of the Web site. Therefore, they will be able to lower the minimum support of the SPM algorithm in the second step of the WUM process and, in this way, to obtain more precise and interesting results.

Based on these issues, in this work, we intend to prove the following two hypothesis:

- An increase in the quality of the data used in a Web Usage Mining process, would allow to obtain better results at the end of the process. Enhancing the quality of the usage data consists in removing unnecessary requests (reducing its size) and structuring the data.

- By adopting divisive approaches for sequential pattern mining it is possible to obtain interesting results when mining large Web access log file, consisting in behaviors of minority groups of users. Currently, such results cannot be directly obtained with a sequential pattern mining technique due to the low support of the sequential patterns. By using a divisive approach, we can extract sequential patterns with lower support values.

In the next section, we introduce the solutions that we propose in this thesis in order to overcome the main WUM issues presented before and to help us validate the hypothesis presented here.

1.5 Thesis Contributions

The Web Usage Mining process is the main subject of our thesis. We are interested in designing and implementing a global approach for this process that will solve most of the current open problems.

The contributions made by this research are the following:

- The first contribution of our research tackles two open problems of WUM preprocessing: the quantity and quality of the data used in WUM. The Web usage data available for analysis can reach several GB per hour (see Chapter 2), therefore, specific preprocessing methods need to be designed to filter unused data and to structure the raw data.

  Our approach provides a **complete preprocessing methodology for WUM** that will allow the analyst to transform any collection of Web server log files into a structured collection of Web requests. Thus, our preprocessing will allow the Intersites WUM.

  We divided the preprocessing stage in four main steps: data fusion, data cleaning, data structuration and data summarization. The log files are first joined in a single file on which the remaining preprocessing steps will be applied. Afterwards, this file is cleaned by removing all unnecessary requests, such as implicit requests for the objects embedded in the Web pages and the requests generated by non-human clients of the Web site (i.e. Web robots). Then, the remaining requests are grouped by user, user sessions, page views, visits and episodes. Ideally, this collection of requests is saved into a relational database with a model that captures the new structure (obtained after the preprocessing) of the Web requests collection.

  The role of this preprocessing is to considerably reduce the large quantity of Web usage data available and, at the same time, to increase its quality by structuring it and providing additional aggregated variables for the data mining analysis that will follow.

- The second contribution of our research for Web Usage Mining concerns the **extraction of sequential patterns with low support from a large structured file of Web usage data**. There are a number of technical issues with the existing sequential pattern mining techniques that prevent their successful application on Web usage data (see Chapter 3). Therefore, we propose three new approaches for extracting the sequential patterns in WUM. Our approaches are able, not only to extract sequential patterns from large structured Web access log files with lower supports than the existing methods, but also to provide **interesting results** on this data. To overcome the data size issue, we propose the use of a divisive methodology that breaks the initial data into smaller groups which can be analyzed easier and quicker afterwards.

  Regarding the interestingness of the results, we argue that finding behaviors common to only a small group of people with similar interest is a surprising, thus interesting result. The behaviors of these “minority groups” are usually hidden inside the log files (due to their low support) and cannot be detected (extracted) with classical methods, as these methods cannot be run with very low support values or the number of patterns extracted would be too high for human result analysis.

- The preprocessing methodology and two of the three methods that we propose in this thesis are also implemented into software tools grouped in the **AxisLogMiner toolbox**. The toolbox will allow any user to conduct a coherent analysis, from the preprocessing step up to the result analysis step.

These contributions are presented in the following chapters of this thesis. A brief summary of these chapters is provided in the next section.

1.6 Thesis Structure

In this section we summarize the content of the remaining chapters of this thesis:

Chapter 2: Methodology for Preprocessing in WUM

In Chapter 2 of the thesis, we present our contributions made to the first step of a WUM process: the Web usage data preprocessing. Our main objective is to reduce in a significant, but relevant manner the size of the Web servers' log files for an Intersites WUM analysis. Moreover, we also intend to increase the quality of the data obtained at the end of the preprocessing step.

We propose a complete methodology for data preprocessing in the case of WUM. This methodology is divided into four main steps: data fusion, data cleaning, data structuration and data summarization. The first step is to be applied only in the case of Intersites WUM. Through data cleaning and data structuration, we significantly reduce the size of the initial log files down to 10-25% of the initial size, meeting in this way our first objective. In the last step, summary data is generated based on the data structure, fulfilling our second objective, the increase in data quality.

To validate the efficiency of our methodology, we conducted an experiment using the log files of INRIA's Web sites: we joined and analyzed together the log files collected from four of INRIA's Web servers. The results showed that our methodology reduced the Web access log files down to 25% of the initial size, while the requests were reduced to 15% of their initial number. In this process, only the unnecessary requests are removed, all the other information is kept and can be recreated from the database that we propose.

We used our data preprocessing methodology to preprocess several datasets. The experiments conducted on these datasets by members of our team or partner teams are also described in this chapter.

Finally, we compare our methodology with the main works in the WUM preprocessing.

Chapter 3: Methods for Sequential Pattern Extraction in WUM

In this chapter we address the problem of discovering useful sequential patterns from a large Web access log: not only the most frequent patterns (usually uninteresting), but also patterns with very low support (more interesting for a specific part of the Web site).

To solve the problem of extracting sequential patterns with low support, we propose a divisive methodology. This methodology splits the initial problem of extracting sequential patterns from a large structured Web access log file into several tasks consisting in extracting sequential patterns from sub-logs generated from the initial log file.

Based on our divisive methodology, we propose three approaches for WUM. These approaches are then instantiated by concrete methods: the "Cluster & Discover" method implements our first approach – the Sequential Approach, the "Divide & Discover" method implements our second approach – the Iterative Approach and the "Hierarchical Discovery" implements our third approach – the Hierarchical Approach.

We conducted several experiments using the Cluster & Discover (C&D) and Divide & Discover (D&D) methods. We show that the C&D method is capable of automatically discovering sequential patterns of very low support, patterns that could not be extracted with other SPM methods such as PSP [MPC99] or WAP-mine [PHMAZ00].

Moreover, interesting browsing behaviors extracted using either C&D or D&D are presented in this chapter. Due to their low support, some of these patterns cannot be extracted using a classical sequential pattern mining algorithm.

At the end of this chapter, we present the main related works in clustering and extracting sequential patterns from Web usage data.

Chapter 4: The AxisLogMiner Toolbox for WUM

In Chapter 4 of this thesis, we present the AxisLogMiner toolbox that we developed to support our methodologies. We designed the AxisLogMiner toolbox as a collection of tools for Web Usage Mining. This software toolbox contains the following tools, whose functionalities and interfaces are presented in this chapter:

1. The **AxisLogMiner Preprocessing** is the software tool that implements our WUM preprocessing methodology. This tool takes as input several log files and outputs the structured sessions into a MySQL database and/or a structured log file. We used *Java* for the application GUI and eight scripts coded in *Perl* for implementing the WUM preprocessing operations.

2. The **Cluster & Discover** tool corresponds to the C&D method described in Chapter 3. The application is able to automatically extract sequential patterns with low support from a structured log file. We used *Java* for the application GUI and the clustering algorithm, *C* for the previously developed PSP algorithm, and scripts coded in *Perl* for splitting the log file.

3. The **Divide & Discover** tool corresponds to the D&D method described in Chapter 3. This application has an interactive mode where the user can decide on what sub-log to iterate the process. As for the C&D application, we also used *Java, Perl* and *C* for implementing this application.

Chapter 5: Conclusions and Perspectives

In the last chapter of our thesis, we summarize the contributions of our thesis and describe the perspectives opened by this research, with the Semantic Web mining being one of the most important topics. Due to the continuous development of the Semantic Web, more information will become available (in a structured format) about the content of Web pages. This information could be used together with the Web usage data to highlight relevant patterns (involving distinct semantic topics) and, therefore, understanding and interpreting these kind of patterns will be much easier.

Chapter 2

Methodology for Preprocessing in WUM

The number of Web resources available on the Internet as well as the number of Web users is continuously growing. As a result, the quantity of the usage data available for a WUM study is also increasing. But this increase in quantity was only recently followed by proposals to improve the quality of the usage data [OBHG03]. The most common Web log format is still CLF/ECLF, designed 10 years ago and this format is not suited to perform data mining analysis, as noted by [HBV04a]. In this chapter we describe a general methodology for preprocessing the raw HTTP logs into a structured form.

A *log file* is a plain text file, where requests are ordered chronologically by the time at which the user requested the resource. Usually, a data mining tool needs records as input, stored as rows in a database table or as transactions (i.e. sequences of items). The aims of the preprocessing step in a WUM process are roughly to convert the raw log file into a set of transactions (one transaction being the list of pages visited by one user) and to discharge the non-interesting or noisy requests (e.g. implicit requests or requests made by Web robots).

When designing this methodology for preprocessing, our main objectives were to reduce the quantity of data being analyzed while, at the same time, to enhance its quality. Some of the steps of this methodology were already proposed by other authors, especially by Cooley [Coo00]. However, we added some new steps and modified some of the existing ones, thus obtaining a complete methodology for preprocessing in WUM, implemented in the AxisLogMiner Preprocessing tool presented in Chapter 4.

First, we formalize the problem of data preprocessing in WUM (Section 2.1). Then, we give an overview of the data preprocessing process (Section 2.2) and we present our four steps of data preprocessing: data fusion (Section 2.3), data cleaning (Section 2.4), data structuration (Section 2.5) and data summarization (Section 2.6). The experiments conducted using our methodology are presented in Section 2.7. A comparison with existing works in WUM preprocessing is made in Section 2.8 and we close the chapter with a discussion and some prespectives for WUM preprocessing in Section 2.9.

2.1 Problem Formalization

Consider the set $R = \{r_1, r_2, \ldots, r_{n_R}\}$ of all Web resources from a Web site. If $U = \{u_1, u_2, \ldots, u_{n_U}\}$ is the set of all the users that have accessed that site, we can define a log entry as $l_i =< u_i; t; s; r_i; [ref_i] >$, where $u_i \in U, r_i \in R, t$ represents the access time, and s represents the request status. ref_i represents the referring page and is optional as in some Web log formats, like the CLF format where the referring page is not recorded. s is a three-digit code indicating the request's success or failure. In the latter case, it also indicates the cause of the failure. A status with a value of 200 represents a succeeded request, while a status of 404 shows that the requested file was not found at the expected location. $L = \{l_1, l_2, \ldots, l_{n_L}\}$, ordered ascending by the time value of l_i, constitutes a Web server log. In the case of N Web servers, the set of log files is $Log = \{L_1, L_2, \ldots, L_N\}$.

The site map is a very useful resource for identifying page views, visits, and episodes during the preprocessing. Cooley formalized a Web site map as:
$\mathcal{M} = [< \mathcal{F}_1; ...; \mathcal{F}_n >], \mathcal{F} = \{h_f, \mathcal{L}_1, ..., \mathcal{L}_m\}, \mathcal{L} =< r, (h_1, g_1)|...|(h_p, g_p) >$,
where $\mathcal{M}$ is the site map and $\mathcal{F}$ is a frame [Coo00]. Each frame is constituted from an HTML file h_f, which is a list $\mathcal{L}_i$ of associated links (h_i) and targets (g_i). r represents the link type (GET, POST, etc.).

The preprocessing groups first the log entries into page views by using the log and the site map, if available. A page view p_i consists of $p_i = \{r_{i_1}, r_{i_2}, \ldots, r_{n_P}\}$, where $r_{i_j} \in R$. The compacted log entry is $lp_i =< u_i, t, p_i, ref_i >$. Considering a time interval Δt, we can define a visit v_i for a user u_i as $v_i =< u_i, t, pv_i >$, where $pv_i =< (t_1, p_1); (t_2, p_2); \ldots; (t_n, p_n) >, t_{i+1} \geq t_i$ and $t_{i+1} - t_i < \Delta t, i = \overline{1..n-1}$.

Using these notations, the preprocessing problem is formalized as follows: *Given a set of log files for several Web sites (Log) and the Web sites' maps $\{\mathcal{M}_i\}$, extract the visits of the Web sites' users for a given Δt.*

2.2 Data Preprocessing Overview

Based on the researches that were already conducted in this domain, we split the data preprocessing step in two main parts: the classical data preprocessing, where we group the methods commonly used in the literature for preprocessing data, and the advanced data preprocessing containing new ideas for data enhancement for the data mining steps that follow.

The *Classical data preprocessing* involves three steps: *data fusion*, *data cleaning*, and *data structuration*. Our solution for WUM preprocessing also adds what we call an *advanced data preprocessing* step (Figure 2.1). This consists in a data summarization step, which will allow the analyst to select only the information of interest. We have successfully tested our solution in an experiment with log files from INRIA's Web sites presented in Section 2.7.

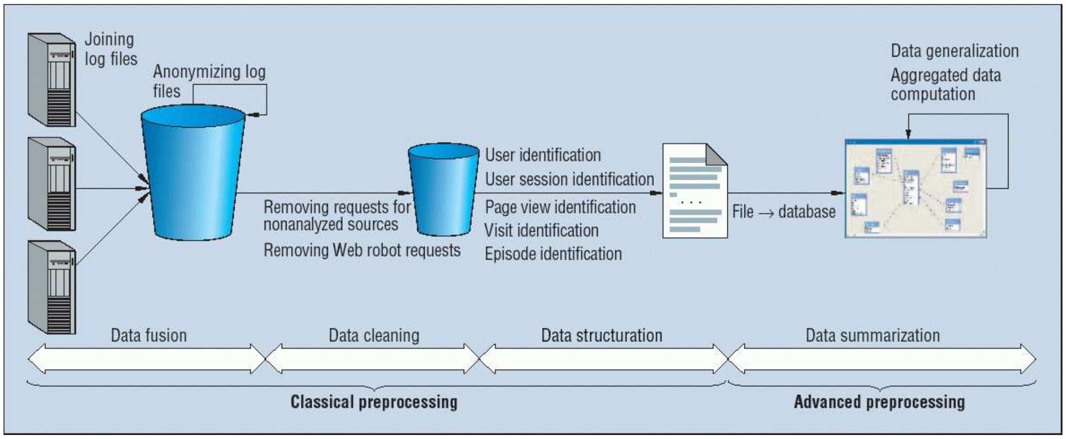


Figure 2.1: The Four Steps of Data Preprocessing [TT04a]

2.3 Data Fusion

At the beginning of the data preprocessing, we have the *Log* containing the Web server log files collected by several Web servers as well as the Web site maps (in XGMML format [PK01], which is an XML application). First, we join the log files and then anonymize the resulting log file for privacy reasons.

2.3.1 Joining Log Files

First, we join the different log files from the *Log*. We put together the requests from all log files into a joint log file. Generally, in the log files, the requests do not include the name of the server file. However, we need the Web server name to distinguish between requests made to different Web servers, therefore we add this information in the requests (before the file path). Moreover, we have to take into account the synchronization of the Web server clocks, including the time zone differences. Figure 2.2 shows our algorithm for joining Web server log files. In this algorithm we used the following notations:

- $Log = \{L_1, L_2, \ldots, L_N\}$;
- L^J is a joint log file;
- $L_i.c$ is a cursor on L_i's requests;
- $L_i.l$ is the current log entry from L_i (indicated by the cursor $L_i.c$);
- $T = \{(id, t) | id = \overline{1, N}, t - temps\}$ is a list containing the IDs of the log files and the time t of the current log entry from L_{id}. The algorithm assumes that the time values t are already synchronized;
- $S = (w_1, w_2, \ldots, w_N)$ is an array with Web server names, where $S[i]$ is the name of the Web server for the log i.

```
Procedure JoinLogs ($\mathcal{L}ogs$)
    $T = \phi$
    for $i = \overline{1, N}$
        $L_i.c = 1$
        $t = L_i.l.time$
        InsertT$(i, t)$
    end for
    while $T.lenght > 1$
        while $T[1].t > T[2].t$
            $id = T[1].id$
            $l' = S[id] + L_{id}.l$
            $L^J = L^J \cup l'$
            $L_{id}.c = L_{id}.c + 1$
            if EOF$(L_{id})$
                RemoveT$(T[1])$
                break
            end if
        end while
        if not EOF$(L_{id})$
            OrderT$(1, T[1])$
        end if
    end while
    while not EOF$(L_{T[1].id})$
        $id = T[1].id$
        $l' = S[id] + L_{id}.l$
        $L^J = L^J \cup L_{id}.l$
        $L_{id}.c = L_{id}.c + 1$
    end while
End JoinLogs
```

Figure 2.2: An Algorithm for Joining the Log Files

2.3.2 Anonymizing Log Files

When sharing log files or publishing results, for privacy reasons, we need to remove the host names or the IP addresses. Therefore, we replace the original host name with an identifier that keeps the information about the domain extension (i.e the country code or organization type, such as .com, .edu, and .org). In our experiment with INRIA's Web sites, we recorded more detailed information including the research unit and the project name. We can also use these parameters later in the analysis, and the log files can be shared without revealing sensitive information.

Example 1 *For instance, the host name* ***mycomputer.mydomain.com*** *is replaced with* ***12345.example.com.com****. In the INRIA experiment, local hosts from our institute were replaced as follows:* ***axis.inria.fr*** *with* ***123456.example.com.11.project.sophia.inria.fr*** *using information about the host's IP address. The first number in these new addresses represents an ID which can be used to retrieve the real name. The string* ***"example.com"*** *is used to mark anonymized host names. For INRIA's hosts we keep also the ID of the team-project (****11****) and the research unit (****sophia.inria.fr****). When instead of a host name we have an IP address, we replace this IP address with a new one, generated randomly to preserve the confidentiality of the data.*

2.4 Data Cleaning

The second step of data preprocessing consists of removing useless requests from the log files. Usually, this process removes requests concerning *non-analyzed resources* such as images, multimedia files, page style files, JavaScript file, etc. Data cleaning also identifies *Web robots* and removes their requests.

For Web portals and popular Web sites, the log file size is measured in gigabytes per hour. For example, as of September 2000, Yahoo, the most popular Web site at that time [Nie00], had collected 48 Gigabytes of log data in one hour [SK02]. In 2003, the total size of log files for Amazon.com was of hundreds of Gigabytes per day [Wei04]. Manipulating such large files is complicated even with the latest hardware tools. By filtering out useless data, we can reduce the log file size to use less storage space and to facilitate upcoming tasks. For example, by filtering out image requests, we reduced INRIA's Web server log files to less than 50% of their original size, as described in Section 2.7.1.

2.4.1 Removing Requests for Non-analyzed Resources

Nowadays, most Web pages contain images. These images either serve for design purposes (e.g. lines and colored buttons) or to present information (e.g. graphics and maps). The decision to keep or remove the log files for these images depends much on the purpose of the WUM. When the purpose is to support Web caching or prefetching, the analyst should not remove the log entries referring to images and multimedia files. For a Web cache application it is more important to predict requests for such files than requests for other (text) files, because the images are usually bigger than the HTML documents. In contrast, when the analysts intend to find the shortfalls in the structure of a Web site or to offer personalized dynamic links to their visitors, they should keep only the explicit requests because these requests represent the real actions of the users.

Image requests removal might require a site map in order to distinguish between explicit and implicit Web requests. Images embedded in the HTML files are included using the IMG tag or by using scripts. To have an explicit Web request for an image, there must be a link to this resource in the HTML file. In conclusion, we must keep these explicit requests as they represent the users' actions.

In addition to image files, we may have other file types that can be included into the Web pages generating implicit requests like page style files, script files (JavaScript), applet files (Java object code) and so on. Except for the resources needing explicit requests (like some applet files), the requests for these files need to be removed as they will not bring any new knowledge in the pattern discovery phase. Finding that a request for a Web page A is followed by a request for its style sheet file will not be an useful pattern although it will be a very frequent one. Therefore, not removing them, will negatively affect the last WUM phase, the pattern analysis.

2.4.2 Removing Web Robots' Requests

A Web robot (WR) (also called *spider* or *bot*) is a software tool that periodically scans a Web site to extract its content. WRs automatically follow all the hyperlinks from a Web page. Search engines, such as Google, periodically use WRs to gather all the pages from a Web site in order to update their search indexes. The number of requests from one WR may be equal to the number of the Web site's URIs. If the Web site does not attract many visitors, the number of requests coming from all the WRs that have visited the site might exceed that of human-generated requests. In our experiment with INRIA's Web sites, WR requests constituted 46% of the total non-image requests.

Removing WR-generated log entries not only simplifies the mining task that will follow, but it also removes uninteresting sessions from the log file. Usually, a WR has a breadth (or depth) first search strategy and follows all the links from a Web page. Therefore, a WR will generate a huge number of requests on a Web site. Moreover, the requests of a WR are out of the analysis scope, as the analyst is interested in discovering knowledge about users' behavior.

Most of the Web robots identify themselves by using the user agent field from the log file. Several databases referencing the known robots are maintained [Kos, ABC]. However, these databases are not exhaustive as each day new WRs appear or are being renamed, making the WR identification task more difficult.

To identify WR hosts, we currently use three heuristics:

1. **Robots.txt (RT):** We look for all the hosts that have requested the page `/robots.txt`. This file contains browsing rules for the WRs that index the Web site, such as the names of the folders not to be indexed. This should be the first file that a WR requests from the Web site. However, obeying the rules from `/robots.txt` is not mandatory and many robots, especially the ones designed for illicit reasons, such as collecting e-mails from Web pages, ignore this file.

2. **Known User Agent (UA):** We use a list of user agents known as robots. The list is created using data from various sources such as [Kos, ABC] and may be completed as new WRs are detected with the other two methods.

3. **Browsing Speed (BS):** We compute the *Browsing Speed* as: *BS = (Number of page views)/(Session duration in seconds)*. If the *BS* exceeds a threshold θ_1 (pages/second), and the number of page views for that visit exceeds another threshold θ_2 pages, we consider the host to be a WR.

Once all the Web robots are identified, we can remove the requests that they generated. This procedure is straightforward, consisting in the removal of all the requests issued by pairs of (**Host, User Agent**) identified as being a Web robot.

2.5 Data Structuration

This step groups the unstructured requests of a log file by user, user session, page view, visit, and episode. At the end of this step, the log file will be a set of transactions, where by transaction we refer to a user session, a visit or an episode.

2.5.1 User Identification

In most cases, the log file provides only the computer address (name or IP) and the user agent (for the ECLF log files). For Web sites requiring user registration, the log file contains also the user login (as the third record in a log entry). In this case, we use this information for the user identification. When the user login is not available, we consider (if necessary) each IP as a user, although we know that an IP address can be used by several users.

2.5.2 User Session Identification

Identifying the user sessions from the log file is not a simple task due to proxy servers, dynamic addresses, and cases where multiple users access the same computer (at a library, Internet café, etc.) or one user uses multiple browsers or computers. However, a number of techniques can provide additional information. To identify a user that already visited the Web site, the most common techniques are cookies, user registration or modified browsers. All these techniques have drawbacks, especially concerning the user privacy.

If the user login is available, we combine the user login field with the pair (**Host, User Agent**) to separate the user sessions. We choose this solution because a registered user might use different computers or browsers when exploring the Web site and the inclusion of the user agent allows us to better distinguish between users within a common host.

Moreover, in an experiment conducted by [BMNS02], the authors report that the combination (**Host, User Agent**) correctly identifies the user in 92.02% of the cases and only a small number of these combinations (1.32%) are used by more than three users (because of proxies). Therefore, we can affirm that using this combination provides a good identification criteria for the user session.

2.5.3 Page View Identification

Using the site map $\mathcal{M}$ (represented with XGMML [PK01]), the requests are grouped by page views with the following algorithm:

- When the request for the page view p_i is in the log file, we remove the log entries corresponding to the embedded resources from p_i, and we keep only the request for p_i.
- When the request for p_i is absent (due to the browser or proxy cache), but some entries for its corresponding resources are present and these entries have p_i in the referrer field, we replace the entries corresponding to the resources with a request for p_i and we set the time of this request to $t_i = min\{time(l_i)\}$, where l_i is the corresponding log entry for the resource r_i.

If the site map is not available, we identify the page views by using the time of the request. For requests made at the same time (i.e. the same second), we keep only the first request (as ordered in the log file) and discard the following ones.

Finally, a third solution consists in using a statistical or a DM approach for identifying Web pages that are usually requested together and in a short period of time. One solution would be to use sequential patterns with high confidence obtained from the user sessions or visits.

After the page view identification, the log file will contain, normally, only one request for each user action.

2.5.4 Visit Identification

So far, we have obtained a sequence of page views for each user. This represents the user's clickstream sequence on the Web site (i.e. the user session) during a certain period. Several heuristics can be used to split the user session into visits:

- H_{Page} uses a threshold Δt for measuring the time gaps between consecutive requests. A new visit begins each time when a gap exceeding Δt occurs between two page views. This is the most common method in WUM and it is also the one we adopted in our preprocessing methodology.

- H_{Visit} uses a time threshold for the entire visit [CMS99, FSS00]. Once the visit duration exceeds Δt, a new visit begins. However, depending on the Web site pages or the users, it may take more time than Δt (usually 30 minutes) to visit the Web site. In this case, using H_{Visit}, the users' visits will be cut and the page views will be separated in different visits.

- H_{Ref} uses the history of the visit and the referrer of the current page view. If there is no link from a previous requested page to the current one, a new visit begins. This heuristic needs the Web site map at the time of the visit, because the Web sites are dynamic and their pages and links are dropped and removed all the time.

- MF (which stands for the "Maximal Forward" algorithm [CPY98]) ends a visit when a backward reference occurs. With the MF algorithm, backward references are dropped. For example, a user that viewed the pages A, B, C, B, D did two visits: A, B, C and A, B, D. This method has a drawback because for some classes of applications, predicting even this kind of "back" reference (e.g. C to B) is important. Because the session is split after the user does a backward reference from C to B, the information about the traversal of the link $C- > B$ is lost at the visit level.

In our methodology, we use the H_{Page} approach with a parameterizable threshold, Δt. In [BMSW01], the authors evaluated the performances of the first three heuristics presented. According to their experiment, the best results for visit identification are obtained either with the H_{Visit} or with the H_{Page} approach. The differences between the two approaches were minor when evaluated in their framework, while the quality of the H_{Ref} approach was low.

2.5.5 Episode Identification

Identifying episodes is a complex problem as it needs: the semantic definition of the entire Web site (i.e. *the Web site ontology*) and a distance measure on the semantic definitions of the Web pages. The *semantic definition* for a Web page represents all the semantic topics associated with the Web page. The *semantic topic* is a label characterizing the content of a Web page. We can have several semantic topics for one page, like in the Figure 2.3 where we represented some of the pages of `www-sop.inria.fr`.

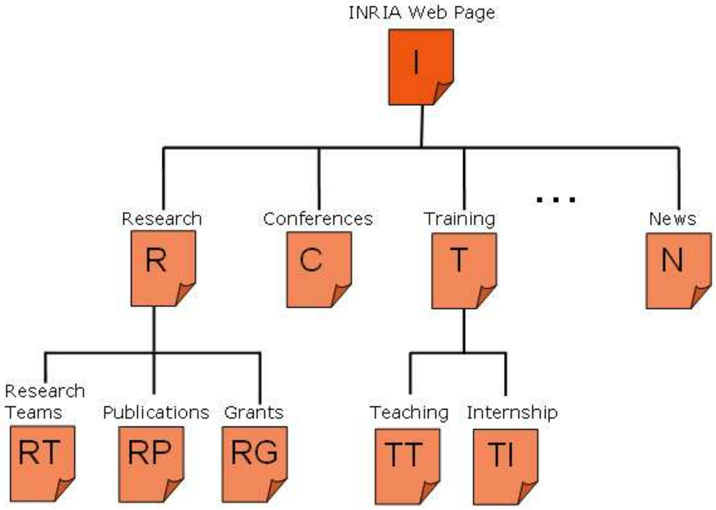


Figure 2.3: **www-sop.inria.fr**'s Semantic Topics Hierarchy

For identifying episodes, we propose the use of a *hierarchy of semantic topics* (see Figure 2.4) and of a *semantic distance* between semantic topics (distance defined on this hierarchy). We propose to identify episodes by calculating the semantic distance between the semantic topics of any two consecutive pages or between the current page and the group of previously visited pages. The distance may be a simple *link distance* counting the number of edges to be traversed from one semantic topic to another. When this distance exceeds a predefined threshold, a new episode begins.

Cooley used different heuristics [Coo00] to determine the type of a page (i.e. syntactic/semantic topic in our case). These heuristics are generally based on the average time the user spends on the page, on the page size, or both, thus they are highly depended on the usage of the Web site and not on its content. Moreover, the number of types proposed was only five, which is very limited compared with the number of topics.

Automatically finding semantic topics for a Web page is still an open problem of Web Mining. One possible solution would be the usage of the XML language to annotate the Web pages with semantic topics when they are created or modified. Then, these

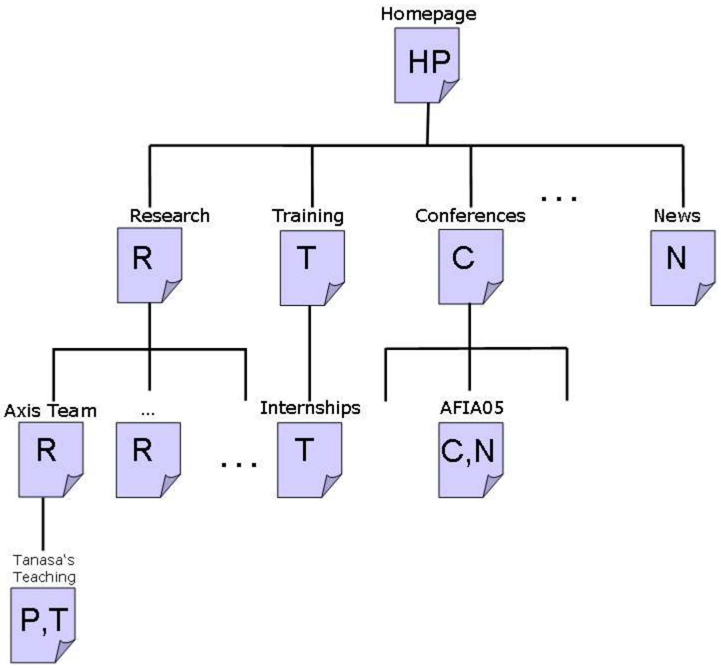


Figure 2.4: Web Pages of **www-sop.inria.fr**

topics must be placed in an hierarchy on which a distance measure can be applied. Recent works in the Semantic Web Mining domain [BHS02, OBHG03], discuss in more detail these issues.

Finally, techniques from Web Content and Text Mining can be used to automatically discover semantic hierarchies for the pages belonging to a Web site [MS01, MBGMF04, CHS04].

2.6 Data Summarization

The last step of data preprocessing contains what we called the "advanced preprocessing step for WUM". In this step, first, we transfer the structured file containing visits or episodes (if identified) to a relational database. Afterwards, we apply the *data generalization* at the request level (for URLs) and the *aggregated data computation* for episodes, visits and user sessions to completely fill in the database. All these operations are described in detail as follows.

2.6.1 Storing the Structured Log File

We designed different tables in the relational database (DB) represented in the Figure 2.5. The following information can be stored in the DB:

- The Web site (URL, semantic and syntactic topics);
- The users that accessed the Web site. This information is structured as described in Section 2.5, i.e. we have one table for each of the entity identified during the data structuration step (users, sessions, visits, episodes, etc.).
- The usage of the Web site (requests, status and type of the requests).

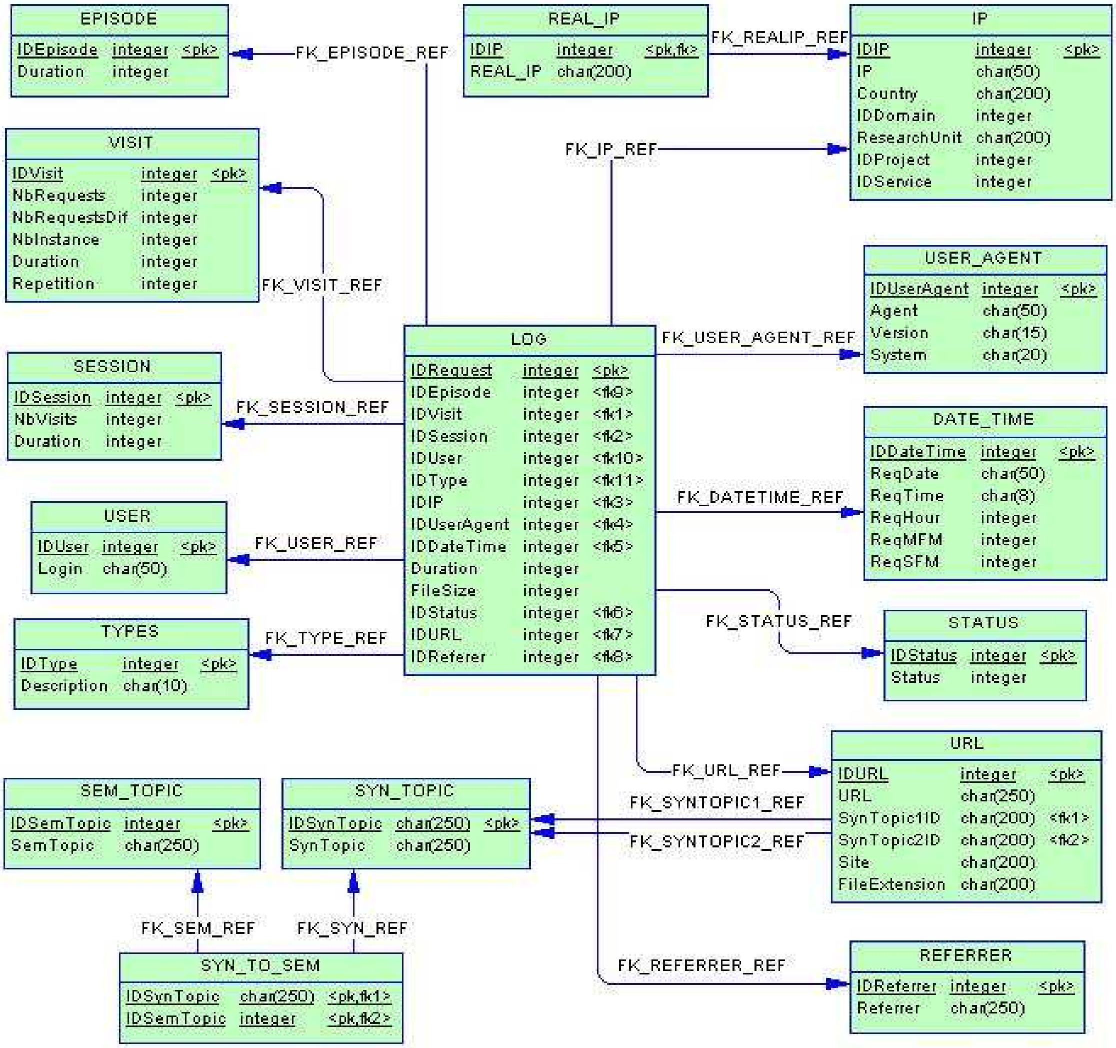


Figure 2.5: Relational Schema for the WUM Database

This model can be further extended by attaching new tables or new attributes to the existing tables. When mining this data, analysts can select only the information in which they are interested.

We describe hereby the main tables of our relational schema for a WUM database.

2.6.1.1 Table LOG

The table LOG is the main table in our relational database (Figure 2.5). Its role is similar to the "Fact Table" from a data warehouse [KR02]. One row of this table corresponds to one request from the original log file. The attributes of this table are mainly IDs (foreign keys) and their aim is to avoid the redundancy of repeating the same field values. The description of the table LOG is given in Table 2.1.

Column	**Description**
IDRequest	ID for the Web request (generated), primary key for the table LOG
IDEpisode	ID for the episode of this request, foreign key from the table EPISODE
IDVisit	ID for the visit of this request, foreign key from the table VISIT
IDSession	ID for the session of this request, foreign key from the table SESSION
IDUser	ID for the user of this request, foreign key from the table USER
IDType	ID for the type of this request, foreign key from the table TYPE
IDIP	ID for the IP initiating this request, foreign key from the table IP
IDUserAgent	ID for the User Agent of this request, foreign key from the table USER_AGENT
IDDateTime	ID for the date and time when this request was done, foreign key from the table DATE_TIME
Duration	Time in seconds between this request and the following one, 0 for the last request of a visit
FileSize	Size of the page requested
IDStatus	ID for the status of the request, foreign key from the table STATUS
IDURL	ID for the URL of the request, foreign key from the table URL
IDReferrer	ID for the referrer of the request, foreign key from the table REFERRER

Table 2.1: Description of the Table LOG

2.6.1.2 Table SESSION

The user sessions are stored in the table SESSION. As mentioned earlier, one user may have several user sessions, if a login is provided and if he used different combinations of **(IP, User Agent)**. Each of these user sessions will generate a row in the table SESSION. The description for this table is provided in Table 2.2

2.6.1.3 Table VISIT

The table VISIT stores the information about the users' visits. It also contains aggregated information about the visits as the number of page views and the repetition factor for a visit. The repetition factor introduced in [Jac98] is computed as:

Column	Description
IDSession	ID for the user session, primary key for the table SESSION
NbVisits	Number of visits that compose this session
Duration	Total duration in seconds for all the visits that compose this session

Table 2.2: Description of the Table SESSION

$$Repetition = 1 - \frac{NbRequestsDif}{NbRequests}.$$

The complete description of the table VISIT is given in Table 2.3.

Column	Description
IDVisit	ID for the visit, primary key for the table VISIT
NbRequests	Number of requests after preprocessing (i.e. page views) for the visit
NbRequestsDif	Number of different requests for the visit
NbInstance	Number of different time instances for the visit
Duration	Time in seconds between the last and the first request of a visit
Repetition	The repetition factor calculated for this visit

Table 2.3: Description of the Table VISIT

2.6.1.4 Table EPISODE

The episodes are stored in the table EPISODE. We may have several episodes for one visit. The description for this table is given in Table 2.4.

Column	Description
IDEpisode	ID for the episode, primary key for the table EPISODE
Duration	Total duration, in seconds, for the episode

Table 2.4: Description of the Table EPISODE

2.6.1.5 Tables IP and REAL_IP

The two tables, IP and REAL_IP, store the description of the users' IPs or the computer names[1]. In the table IP (described in Table 2.5), the IP address is kept in an anonymized form together with some extra information (e.g. country, organization). The link between the anonymized form and the real IP is done via the table REAL_IP which

[1] In this section we will continue to use IP for referring to both the IP or the computer name of a request.

contains the IP in the initial form. The last three columns of the IP table were used in the experiment we conducted on INRIA's Web log files. These attributes provide more details about the user and they can be adapted to other organizations as well.

Column	**Description**
IDIP	ID for the IP, primary key for the table IP
IP	Anonymized IP (the real IP is kept in the table REAL_IP)
Country	Geographical information about this IP (used only for computer names)
IDDomain	When anonymized, the computer names receive a domain ID, stored in this column
ResearchUnit	This field was used for INRIA's computers to designate the research unit of this IP
IDProject	Internal ID for INRIA's project to which this IP belongs
IDService	Internal ID for INRIA's service to which this IP belongs

Table 2.5: Description of the Table IP

2.6.1.6 Table USER_AGENT

The table USER_AGENT contains the description of the browsers used to make the requests. The data used to fill in this table is extracted from the User Agent field of the ECLF log format, when this field is present. However, there is no standard format for this field and various heuristics need to be employed to determine the operating system, the browser type and its version. The complete descriptions of the columns included in the table USER_AGENT are given in Table 2.6.

Column	**Description**
IDUserAgent	ID for the user agent, primary key for the table USER_AGENT
Agent	The string corresponding to the user agent
Version	Version of the browser
System	Operating system on which the browser was running

Table 2.6: Description of the Table USER_AGENT

2.6.1.7 Table DATE_TIME

We created a separate table to store the date and the time of the requests as several requests may have the same date and time. This also allowed us to have more details regarding the time as described in Table 2.7.

Column	Description
IDDateTime	ID for the date and the time, primary key for the table DATE_TIME
ReqDate	The date when the request was made (dd/mm/yy)
ReqTime	The time when the request was made (hh:mm:ss)
ReqHour	Hour of the request (0 to 24)
ReqMFM	Time in minutes from midnight (e.g. 130 for 02:10:05)
ReqSFM	Time in seconds from midnight (e.g. 7805 for 02:10:05)

Table 2.7: Description of the Table DATE_TIME

2.6.1.8 Tables URL, SYN_TOPIC, SEM_TOPIC and SYN_TO_SEM

In the first three tables, the URL, SYN_TOPIC and SEM_TOPIC, we store information about the URLs of the Web site. The main table, the table URL, contains the URL and the ID of the syntactic topics of level 1 and 2 (detailed in Section 2.6.2). The syntactic topics are stored in the table SYN_TOPIC while the semantic topics are stored in the table SEM_TOPIC and are related via the table SYN_TO_SEM. The complete description of the table URL is given in Table 2.8.

Column	Description
IDURL	ID for the URL, primary key for the table URL
URL	The URL as taken from the log file
SynTopic1ID	The ID for the syntactic topic of level 1, foreign key from SYN_TOPIC
SynTopic2ID	The ID for the syntactic topic of level 2, foreign key from SYN_TOPIC
Site	Web site name for Intersites WUM
FileExtension	Extension of the file in the URL

Table 2.8: Description of the Table URL

2.6.2 Data Generalization

In the data generalization process we obtain the generalized versions of the URLs at the chosen abstraction levels, based on the URLs' *syntactic* or *semantic* topics. The objective of this process is to efficiently reduce their number, for the data mining analysis that follows.

The *syntactical generalization* (at level k) for an URL refers to the reduction of the URL depth to k. For example, if we consider the following URL of depth 5: `http://www.mysite.com/L1/L2/L3/L4/page.html`, we can syntactically generalize this URL at level 1 and we obtain `http://www.mysite.com/L1` (L1 is considered as a *syntactic topic* of level 1).
The *semantical generalization* is based on the semantic hierarchy of the Web pages instead of the site topology. The *semantic topics* are organized as a hierarchy –

Site	`www-sop.inria.fr`	`www-sop.inria.fr`	**Total**
Web Pages	75,316	189,151	264,467
Level 1	102	132	234
Level 2	8,332	16,242	24,574
Level 3	25,862	34,502	60,364

Table 2.9: Number of Web Pages and Syntactic Topics at Different Levels

in the same way as an ontology. Generalizing the URLs from a semantic point of view means using a higher semantic topic for this URL with respect to the hierarchy. For instance, if we consider the URL from the previous paragraph, we may replace `L1/L2/L3/L4/page.html` with the semantic topic "research team" (considering that this page belongs to a research team).

In the past, Fu et al. [FSS00] used syntactical generalization to simplify URLs. For example, they reduced the URL `www.umr.edu/~regwww/ugcr97/ee.html` to `www.umr.edu/~regwww/ugcr97/` and then to `www.umr.edu/~regwww/`. The sessions generalization reduces significantly the dimensionality of the data as shown in Table 2.9. Another advantage of the syntactic generalization is that the future updates of the Web site (Web pages addition and deletion) will not affect the hierarchy used as long as the higher structure remains the same [FSS00].

However, the syntactical approach has several drawbacks. Sometimes, a folder (representing a syntactic topic) may group files with different contents, for the ease of Web site maintenance and update. Therefore, this approach will group together pages with unrelated contents located in the same folder (of the Web server). Moreover, Web pages with the same semantic topic (related content), but situated on different locations on the Web site structure will be grouped under different generalized URLs.

A better alternative to the syntactical approach is the semantical generalization that we defined and adopted in our preprocessing methodology. We group pages with similar content under the same semantic topic (i.e. the same semantic topics used for episode identification and listed in Appendix A). However, in our experiment, INRIA's Web sites had tens of thousands of distinct pages, and manual classification was too time-consuming. Thus, we mapped only the first- and second-level syntactic topics under the semantic topics. For example, the URL `www-sop.inria.fr/axis/software/index.html` has "axis" as syntactic topic at level 1, "software" as syntactic topic at level 2. The semantic topic of level 1 for this URL, "projets" (research teams) is implicit and, therefore, it is not present in the address. The semantic topic of level 2 for this URL is "axis".

2.6.3 Aggregated Data Computation

The last step of the data summarization concerns the computation of aggregated variables at different abstraction levels (e.g. request, visit, user session). These aggregated variables are later used in the data mining step as shown in Section 2.7.2. They represent statistical values that characterize the objects analyzed. For instance, if the object analyzed is a **user session**, in the aggregated data computation process, we propose to calculate the following variables:

- The number of visits for that session;
- The length of the session in seconds (the difference between the last and the first date of the visit) or in pages viewed (the total number of page views);
- The number of visits for the period considered, which can be a day, a week, or a month;
- The percentage of the requests made to each Web server.

The first two variables are stored as attributes in the SESSION table (Table 2.2). The other two variables can be calculated from the information available in the database.

If the object analyzed is a **visit**, then we propose to compute the following variables:

- The length of the visit in terms of time and page views;
- The repetition factor of the visit, calculated as described in Section 2.6.1.3;
- The percentage of successful requests;
- The average time spent on a page by the user during the visit.

These are only a few examples of simple aggregated variables that can be computed during the last step of the preprocessing (the first two variables being calculated in this step are stored in the table VISIT). Depending on the objective of the analysis, the analyst can decide to compute additional and more complex variables. These new variables can be calculated by means of SQL and stored in an extended version of our relational model. Another solution consists in extending the preprocessing tool, the AxisLogMiner Preprocessing, that implements our methodology. We present a brief description of our software application in Section 4.1.

2.7 Experiments

We applied our preprocessing methodology on several log files collected from INRIA and UFPE[2] Web sites to prepare the raw log files for the data mining step. Through these experiments we showed that our preprocessing methodology reduces significantly the size of the initial log files by eliminating unnecessary requests and it increases their

[2] Federal University of Pernambuco from Recife (Brazil)

quality through better structuring and by calculating new aggregated variables (used in the data mining step). Moreover, the Intersites feature offered by our methodology can be used to reconstruct sessions over multiple Web servers for an overall analysis of the organization's Web sites.

In this section, first we present the detailed results of the preprocessing step applied on the log files gathered from two of INRIA's Web sites in order to show that our preprocessing methodology allowed us to significantly reduce the size of the initial dataset. Second, to evaluate the quality of the preprocessed data, we describe two clustering analysis realized based on aggregated variables computed by our methodology. Third, we present two analysis on the users' visits made to both `www.inria.fr` and `www-sop.inria.fr` Web sites. Finally, we list the additional WUM analysis that were conducted by members of our team and partner teams using this preprocessing methodology. For all the analysis, we provide a short summary of the datasets employed and the analysis carried out.

2.7.1 Data Reduction for INRIA's DS_1 Dataset Preprocessing

In this section, we present only the results obtained by preprocessing the log files of INRIA's DS_1 dataset. We detail the size reduction that we achieved with our methodology for each preprocessing operation. Results of the knowledge extraction steps on similar datasets from INRIA's Web sites are presented in the following sections (2.7.2 to 2.7.4).

2.7.1.1 Data Fusion and Cleaning

We ran the tests on a dataset DS_1 of log files gathered in three months (from November 2001 to January 2002) from four of INRIA's Web sites: `www.inria.fr`, `www-sop.inria.fr`, and two search engines of these Web sites[3]. The initial size of the dataset was 4,432 MB. The first four rows of the Table 2.10 list the size of the log files that we obtained after the data fusion step.

Log \ Month	**Nov. 2001**	**Dec. 2001**	**Jan. 2002**	**Total**
Logs for `www.inria.fr`	897	896	1,069	2,862
Logs for `www-sop.inria.fr`	507	449	598	1,554
Logs for two INRIA search engines	3	7	6	16
Total logs	1,407	1,352	1,673	4,432
Logs cleaned (images only)	602	680	760	2,042
% of initial size	42.8	50.3	45.4	46.1

Table 2.10: Size of Log Files (in MB) Before and After Removing Non-analyzed Resources

[3] The Altavista-based search engines had the addresses `indexation.inria.fr:8080` and respectively `indexation.inria.fr:8081`.

The last two rows of the Table 2.10 show the results of removing non-analyzed resources. In this step, we eliminated requests for image files, i.e. all the requests for the files with the extensions belonging to the following list: `.jpg,.jpeg,.gif,.png` and `.ico`. This step reduced the merged log file to 46.1% of the initial size of the data set.

Table 2.11 shows how many requests coming from Web robots were removed with each of the three methods presented in Section 2.4.2. As the three methods have non disjunctive results, the same request may be identified by more than one method. The table shows the reduction obtained by applying each method separately and by applying all three methods subsequently.

Heuristic \ Month	**Nov. 2001**	**Dec. 2001**	**Jan. 2002**	**Total**	**Total(%)**
Initial size	602	680	760	2,042	100
Robots.txt	416	430	518	1,364	66.9
Known User Agent	535	590	691	1,816	88.9
Browsing Speed	529	605	691	1,825	89.4
All three heuristics	314	339	440	1,093	53.5

Table 2.11: Size of Log Files (in MB) Before and After Removing Web Robots' Requests

As Table 2.11 shows, removing the Web robot requests reduced the merged log file to 53.5% of the size obtained after removing the non-analyzed resources. The results obtained with the three methods employed for Web robot detection overlapped as shown in the Figure 2.6.

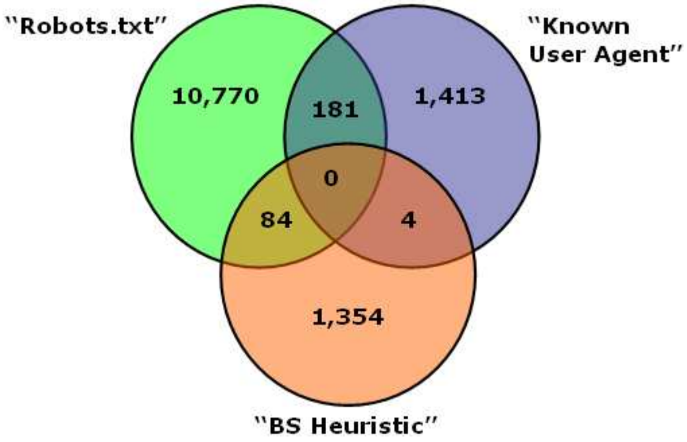


Figure 2.6: Number of Web Robot Hosts Identified Using Each Method

At the end of the data cleaning step we obtained a log file of 1,093 MB instead of the initial 4,432 MB (see Table 2.12), which means that we reduced the size of the file to

24.66% of its initial size. However, in terms of Web requests, the reduction was more significant as we initially had 22,172,327 requests and we obtained 3,485,239 after the data cleaning step. Therefore, the data cleaning reduced the number of requests to 15.7% of their initial number. The percentage difference between the file size and the number of the requests resulted from the supplementary/new information we added or we modified for each request (the Web server and the anonymization details). This information increased the average length of the requests.

Dataset	**Web site(s)**	**Period**	**Size**	
			Before	**After**

DS_1	`www.inria.fr`
	`www-sop.inria.fr`
	`indexation.inria.fr:8080`
	`indexation.inria.fr:8081`
		01.11.2001 – 31.01.2002	4,432MB	1,093MB

Table 2.12: Data Reduction for Dataset DS_1

2.7.1.2 Data Structuration

In the data structuration step, we determined that the 3,485,239 requests obtained after data cleaning were made by 435,434 users (i.e. IP addresses). For INRIA's Web server log files, we did not have the user login information, therefore, we used only the couple **(IP, User Agent)** to identify the 548,832 user sessions. As the site map was not available at the time of the experiments, we used our second method, described in Section 2.5.3, to identify a total of 2,985,639 page views. We separated the users' sessions into 855,459 visits using the H_{Page} heuristic with a threshold Δt equal to 30 minutes, widely used as a standard in the WUM research. 48.9% of these visits were one-page visits and therefore we ignored them later in the data mining analysis.

Preprocessed Requests	**Users**	**Sessions**	**Visits**	**Episodes**	**Page Views**
3,485,239	435,434	548,832	855,459	1,273,893	2,985,639

Table 2.13: Results of Data Structuration for DS_1

We defined 55 semantic topics for the 100 most used syntactic topics of level 1 for both Web sites. The list of the semantic topics as well as the correspondence tables are given in Appendix A. However, considering the huge number of URLs available in the two Web sites[4], defining a hierarchy of semantic topics for deeper syntactic levels required too much manpower. Therefore, we used only one level of semantic topics and we considered that a new episode starts each time that the semantic topic of a

[4] We counted 264,467 distinct and valid requests for pages on the two Web sites, during the 3-month period considered.

page view from a visits changes. Using this heuristic, we obtained a total of 1,273,893 episodes.

2.7.1.3 Data Summarization

The two Web sites studied (`www.inria.fr` and `www-sop.inria.fr`) had 264,467 pages when we collected the log files. For these pages, we determined 55 different semantic topics listed in Appendix A.

We computed the *aggregated variables* described in Section 2.6.3: number of visits per session, length of session (seconds), length of visit (page views, time) and visit repetition factor.

The database obtained was analyzed using SAS software by IUT Menton's students for their Statistics project in 2002. They performed various statistical analysis like Principal Component Analysis (PCA) and Multiple Component Analysis (MCA) to find correlations on the data.

2.7.2 Enhancing Data Quality for Clustering Analysis

In order to show how the quality of the preprocessed data improves the result of the data mining analysis, we choose to present a work conducted with members of our team [ALT$^+$03] on another dataset (DS_2) collected from INRIA's Web sites. In this work, we investigated the relations between the Web site structure (using the syntactic and semantic topics) and the Web site usage (using the log file). We proposed two clustering methods based on PCA and MCA and applied them on preprocessed Web log data. The clustering methods proposed used the aggregated variables calculated in the last step of our preprocessing methodology.

The purpose of the first clustering analysis was to detect groups of similar visits. For this, we used the relational database to select visits respecting the analyst-defined criteria (e.g. visits longer than 60 seconds) and we chose several aggregated variables to be used as visit descriptors (see Section 2.7.2.3). We were able to select and calculate our additional variables without difficulty due to the fact that the data was already structured and stored in a relational database. We consider that the valuable results obtained with our methods were also largely due to the data quality in the first place.

The second analysis (described in Section 2.7.2.5) was focused on the Web site and, therefore, we grouped together semantic and syntactic topics, according to their usage. This analysis used qualitative variables as input for the clustering method.

2.7.2.1 Preprocessed Dataset

The Web logs were collected for a period of two weeks, between the 1^{st} to the 15^{th} of January 2003. The details about the datased DS_2 are presented in Table 2.14.

From the Table 2.15, we can see that, in this case, the reduction of the data was even

Dataset	**Web site(s)**	**Period**	**Size**	
			Before	**After**

DS_2	`www.inria.fr`
	`www-sop.inria.fr`
	`www-futurs.inria.fr`
		01 – 15.01.2003	1,116MB	139MB

Table 2.14: Details of Dataset DS_2

more significant than in the previous experiment on dataset DS_1 (Section 2.7.1), as the number of requests remaining after the preprocessing step represents 11.1% of the initial number of requests.

Requests	**Preprocessed Requests**	**Users**	**Sessions**	**Visits**
6,040,312	673,389	104,024	115,825	174,015

Table 2.15: Results of Data Structuration for DS_2

According to our relational model represented in Figure 2.5, we have three types of information in the database:

- Information about the Web sites usage (requests, visits, sessions, etc.),
- Information about the users (IP, domain name, country, etc.) and
- Information about the Web sites (URL, syntactic and semantic topics, etc.).

The details about the first type of information are provided in Table 2.15 and we present the other two types below.

Information about the users

During the period analyzed, a total of 104,024 users (i.e. distinct IP addresses) made requests to the three Web servers. Most of the users came from France (30.7%) and from those, 1,989 users (1.7%) came from one of INRIA's research units. If the research unit was "Sophia Antipolis" (i.e. 429 users) then we also knew the research team or service to which the user belonged. The users made 115,825 user sessions (distinct IPs and User Agent) and 174,015 visits. We did not identify episodes because, for our analysis, we had to include multiple semantic topics in one transaction (i.e. visit).

Information about the Web site

There were 86,640 URLs requested from the three Web sites. From these, only 73,427 requests were successful (code between 200 and 300). According to their syntax (i.e. their syntactic topic of level 1), the URLs were grouped into 55 semantic topics.

We used 55 first level semantic topics defined for the first experiment (Section 2.7.1) for both Web sites (`www` and `www-sop`). As a reminding, to each 1^{st} level syntactic topic corresponds a 1^{st} level semantic topic. For example, the 1^{st} level syntactic topic for the URL `http://www-sop.inria.fr/axis/software/index.html` is `‘‘axis’’` and the 1^{st} level semantic topic is `‘‘projets’’` (research teams). The list of these semantic topics and the mappings between syntactic and semantic topics for both Web sites are provided in Appendix A.

2.7.2.2 Data Selection

Another advantage of our relational model is that it allows to easily select a subset of the preprocessed data according to the analyst-defined criteria.

For the first analysis (as described in Section 2.7.2.3), we decided to keep only the "long" visits (in terms of time and number of requests). Therefore the first criteria was the duration of the visits, which had to be superior to 60 seconds. The second criteria was that the number of pages in a visit had to be superior to a threshold of 10 pages. Finally, the third condition was that the "Browsing Speed" (i.e. the ratio between the duration of the visit and the number of pages visited) had to be superior to a threshold of 0.25 pages/second (15 pages/minute). After applying all these constraints to our database of visits, we selected 9,700 visits. The number of valid requests (i.e. the status code was between 200 and 399) for these visits was 282,705. On this set of visits we applied the first clustering technique proposed (see Section 2.7.2.3).

For the second analysis (as described in Section 2.7.2.5), we chose from the first set of visits only those visits that had requests on both `www` and `www-sop` Web sites. The number of visits containing requests for pages on the `www-futurs` Web site was very small (28), therefore we decided not to include them in this analysis. Finally, the number of visits selected for the second analysis was 3,969.

2.7.2.3 Clustering Method for Numeric Web Usage Data

To show the benefits of our preprocessing methodology, we decided to cluster the objects (entities) identified in the data structuration step using variables calculated through our data aggregation operations.

The first analysis method is a hybrid clustering method for the numeric Web usage data: the objects analyzed are the "visits" and the variables have continuous values. This hybrid clustering combines the Principal Component Analysis with the Dynamic Clustering Method (DCM) [Did71]. First, the PCA is used for visualizing the correlations between the variables defined on the log file (in the Data Summarization step). Next, the DCM is applied on the principal factors identified by the PCA.

The objective of the DCM is to find a group of homogenous "visits". The DCM algorithm is a k-means-like algorithm which uses the Euclidean distance. The number of clusters is empirically determined by the SPAD software used to cluster the data. The algorithm can be summarized as follows:

Step 1: Initialization
Let k objects of a set of random "visits" be the "representation" of the k clusters. Let us call them A_1,...A_k.

Step 2: Assignment Step
All the visits X_i are assigned to the cluster k iff $d(X_i, A_k)$ is minimum.

Step 3: Representation Step
A new representation A_k is computed as the average of the elements contained in the cluster k.

Step 4: Stability
If no visit changes a cluster then stop, else go to step 2.

The six active variables describing a visit (presented in Table 2.16) are calculated in the advanced data preprocessing step of our methodology. Three of these variables (NbRequests, Duration and Repetition) are already present in our relational model, as columns in the table VISIT (see Table 2.3). The other three variables are calculated using SQL and the information existing in our database (the tables LOG, VISIT and STATUS). These variables are used afterwards in the PCA analysis.

Variable	**Description**
PRequest_SEL	Ratio of successful requests
NbRequests	Total number of requests
Duration	Total duration of the visit
Repetition	Ratio related to the requests repeated
MDuration_OK	Average duration of a request inside the visit
MSize_OK	Average size of a page reached

Table 2.16: Description of the Active Variables

The PCA analysis identifies the principal factors based on the six variables that describe each visit. Afterwards, we apply on these factors the DCM algorithm described earlier.

Finally, the visits clustering phase is followed by the description of the clusters and of their positions on a factorial plan.

In the following section, we present the results that we obtained by applying this method on the dataset DS_2.

2.7.2.4 Results of Numeric Variables Clustering

By using the PCA, we analyzed the variables described in Table 2.16. Then, we clustered the visits based on the principal factors identified. The results obtained in the PCA are shown in the correlation circle (Figure 2.7). From this figure we can see that the number of requests, the number of successful requests and the repetition factor are correlated (`NBRequest`, `NBRequest_OK` and `Repetition`). However these variables are not correlated with the percentage of successful requests (`PRequest_SEL`), which means that having a high number of requests increases the risk of following broken links.

We obtained 7 different clusters (shown in the clusters' map from Figure 2.8), but the clusters 1, 2, 4 and 5 were the most significant ones. The largest cluster was Cluster 1 containing 61.4% of the visits. The average number of page views per visit (21 pages) was lower than the average value obtained for all 9,700 visits (54 pages). Also, the average duration for a request (26.1 seconds) was lower in this cluster than the average obtained for the entire dataset (58.4 seconds). We concluded that this cluster may contain visits from the users that are familiar with the analyzed Web sites.

Another interesting cluster, although very small (only 17 visits), was the cluster number 6. It contained very long visits (with an average of 887.76 pages/visit), coming from the `.net` domain extension, usually made at night and on the 1st of January. We concluded that these visits were made by Web robots because it is known that these tools generally crawl Web sites at night and, usually, at the beginning of the month. These WR sessions were not detected in the data cleaning step as they did not verify any of the three heuristics presented. However, with a different threshold for the third heuristic (Browsing Speed) we can also detect these WRs.

2.7.2.5 Clustering Method for Qualitative Web Data

The second analysis is also a hybrid clustering method, but it was applied on qualitative variables. It consists in a Multiple Correspondence Analysis (MCA) on the selected visits and a Dynamic Clustering Method (DCM) on the results of the MCA. The input of the MCA consists in a set of qualitative variables corresponding to the semantic topics defined for the two Web sites (please refer to Appendix A). After the MCA step, we use a DCM with a χ^2 metric to obtain groups of visits that are interpreted afterwards.

The algorithm used for this purpose is Diday's *moving clouds* method [Did71], belonging to the family of the k-means algorithms. This algorithm seeks for a partition P of E consisting in k clusters and an array L of k prototypes so that it minimizes: $\Delta(P^{\bullet}, L^{\bullet}) = Min\{\Delta(P, L) \mid P \in P_k, L \in D_k\}$, where P_k is the set of E's partitions in k non empty clusters. The criterion D expresses the similarity between the partition P and the array of k prototypes. It is often defined as the sum of the distances between all the objects s of E and the prototype g_i of the cluster C_i – the nearest of them.

For this method, we defined two new visit parameters for each Web server considered:

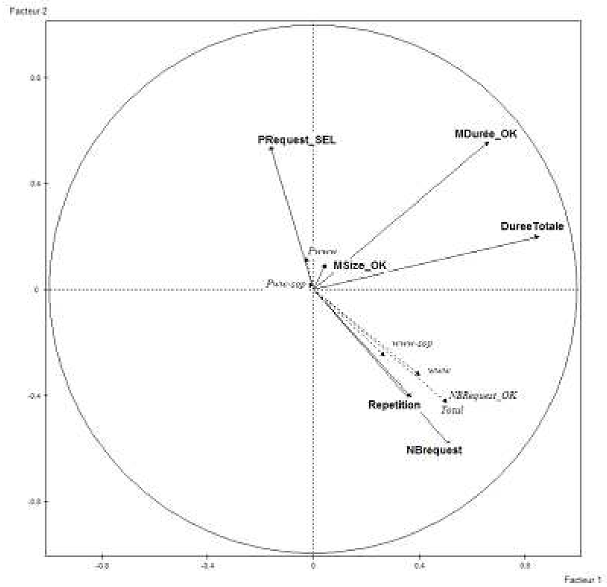


Figure 2.7: The Correlation Circle

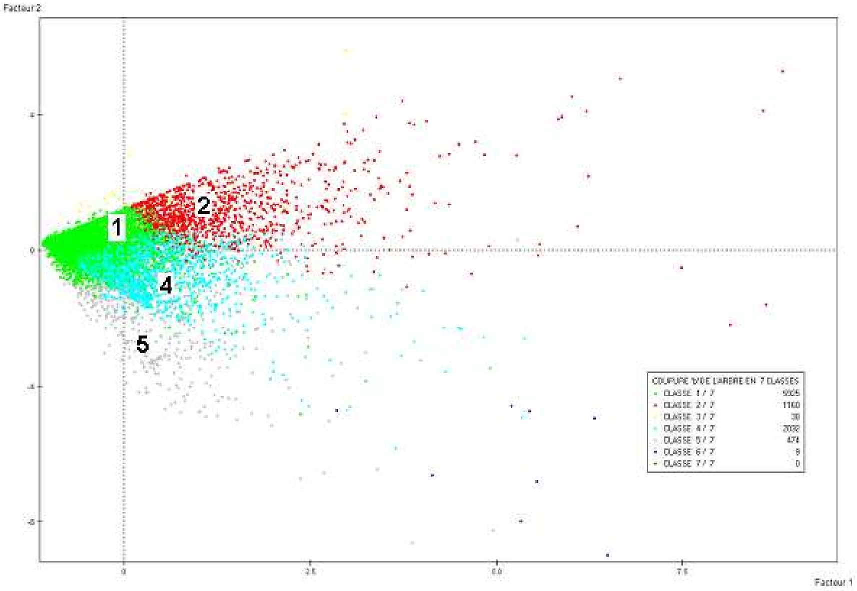


Figure 2.8: The PCA Result

SyntacticTopic-WebServer – Vector of syntactic topic counts for the WebServer
SemanticTopic-WebServer – Vector of semantic topic counts for the WebServer

The four vectors (two for `www` and two for `www-sop`), counted for each visit the number of Web pages corresponding to each syntactic or semantic topic of level 1. They were subsequently used as input in the clustering algorithm.

2.7.2.6 Results of Qualitative Variables Clustering

By applying the *moving clouds* method on the results of the MCA, we obtained 11 clusters. For example, the cluster number 10 (Table 2.17) represents 19.01% of the visits. This cluster contains the visits to the Web pages of INRIA's reporting activity (e.g. activity reports, technical reports and other reports). In this case, the goal of these users is precise: the visitor interest is to read the activity reports of INRIA's researchers. This real life application emphasizes the relevance of our methodology.

Label	**V.Test**	**Cluster/Freq**	**Freq/Cluster**	**Global**
`www ra`	240.13	85.27	66.49	14.82
`www rrrt`	12.85	28.82	3.00	1.98
`www rapports`	10.68	44.13	0.56	0.24
`sop rapports`	7.17	35.31	0.45	0.24

Table 2.17: Description of Cluster 10

2.7.3 Intersites WUM Analysis

We evaluated the Intersites feature of our preprocessing methodology on two different datasets from three of INRIA's Web servers. The first dataset is DS_2 (presented in Table 2.14) and the second one is DS_3 (presented in Table 2.18).

Dataset	**Web site(s)**	**Period**	**Size**	
			Before	**After**
DS_3	`www.inria.fr`	01 – 15.01.2003 &	1,116MB	139MB
	`www-sop.inria.fr`	27.05 – 10.06.2004	901MB	135MB

Table 2.18: Details of Dataset DS_3

Applying our preprocessing methodology on these datasets allowed us to structure the Web requests by identifying the users, user sessions and visits. The figures for the dataset DS_3 is presented in Table 2.19.

The preprocessed log files were subsequently used by members of our theme for data mining analysis. In these experiments only visits containing requests to two of INRIA's Web sites, `www.inria.fr` and `www-sop.inria.fr`, were selected. The intersites aspect

Dataset	**Requests**	**Users**	**Sessions**	**Visits**
DS_3	1,320,895	188,923	206,897	305,304

Table 2.19: Results of Data Structuration for DS_3

of the data allowed the analysts to have an overall view of how the two INRIA's Web sites are used. They were mainly interested in verifying that the Web sites' syntactical and semantical structures correspond to the Web sites' usage.

2.7.3.1 Two Intersites WUM Analysis on DS_2 [Gol04, LV04]

The dataset DS_2 was used in two different Intersites WUM analysis summarized in Table 2.20. We indicate for each analysis the objects analyzed as well as the selection criteria that were applied on the objects of the dataset. The number of the objects selected is also given together with the analysis method used.

Ref.	**Objects**	**Criteria**	**Number**	**Analysis**
[Gol04]	Visits to both `www.inria.fr` and `www-sop.inria.fr`	Status = 2xx Duration $\geq$ 60s	3,696	Batch Self-Organizing Maps (SOM)
[LV04]	Visits to both `www.inria.fr` and `www-sop.inria.fr`	$BS \leq 0.25$ Duration $\geq$ 60s NbRequests $\geq$ 10	2,639	Dynamic Clustering

Table 2.20: Summary of DS_2 Intersites WUM Analysis

Batch SOM Analysis [Gol04]

In her thesis [Gol04], El Golli used an adapted version of the Batch Self-Organizing Maps (Batch SOM) [Koh01]. She applied this algorithm to extract clusters of visits on the 3,969 selected visits. The criteria used to select these visits were, as mentioned previously:

- The visit had requests on both `www-sop.inria.fr` and `www.inria.fr`,
- The duration of the visit was superior to 60 seconds and
- The number of successful requests was at least 10.

The output of the algorithm consisted in 20 clusters and, for each of them, one representative individual was described by its first level syntactic topics. The author showed that 16 out of the 20 cluster representatives had common topics with neighbor clusters; thus, the clustering permitted to identify similar visits and related clusters of visits.

The second analysis carried out clustered syntactic topics to find the correlations between them. The author used the Jaccard similarity [SS73] to indicate the probability that two syntactic topics were visited together in the same visit. She showed that the clusters obtained for the syntactic topics representing INRIA's research teams respected INRIA's internal organization based on research themes.

Dynamic Clustering Analysis [LV04]

In [LV04], the authors proposed a crossed clustering algorithm in order to partition a set of symbolic objects in a predetermined number of classes and, at the same time, to determine a structure (taxonomy) on the categories of the object descriptors. The procedure is an extension of the classical simultaneous clustering algorithms proposed on binary and contingency tables. Their approach is based on a dynamical clustering algorithm applied on symbolic objects. The criterion optimized is the chi-square distance computed between: the description of the objects using modal variables (distributions) and the prototypes of the classes, represented by marginal profiles of the objects set partitions. The convergence of the algorithm is guaranteed and the results are expressed as: the best partitions of the symbolic objects in r classes and the categories of the symbolic descriptors in c groups, respectively.

The results obtained by the algorithm proposed can be considered as a simple example of an automatic clustering procedure that structures complex data and, at the same time, detects typologies of navigation and groups of topics that are homogenous from a semantic point of view. An application on the visits selected from DS_2 (see Table 2.20 for selection criteria) allowed them to validate the proposed procedure and to suggest it as an effective tool in the Web Usage Mining framework.

2.7.3.2 Hierarchical Clustering Analysis on DS_3 [CT04]

Two periods were analyzed comparatively in [CT04]. The dataset contains the logs for two periods of 15 days (1 to 15 January 2003 and 27 May to 10 June 2004), collected from `www.inria.fr` and `www-sop.inria.fr` Web sites. They were preprocessed with our methodology presented previously. For the analysis purposes, the authors selected a reduced set of visits that fulfilled the criteria listed in Table 2.21. The two Web sites were reorganized in April 2004 to match the new research teams structure. The purpose of the research study was to check whether this change influenced the users behaviors when browsing the Web site.

Ref.	**Objects**	**Criteria**	**Number**	**Analysis**
[CT04]	Visits to both `www.inria.fr` and `www-sop.inria.fr`	Status ≤ 400, $BS < 0.25$ Duration > 60s NbRequests > 10	7,418	Hierarchical Clustering

Table 2.21: Summary of DS_3 Intersites WUM Analysis

The authors used their 2-3 Agglomerative Hierarchical Classification (2-3 AHC) algorithm [CBT04] and studied the repartition of the research team topics in the clusters obtained. They noticed that the research teams distribution usually corresponded to their "theme" membership[5] (16 out of the 19 clusters obtained contained teams from the same theme).

In the second analysis conducted, they studied the impact of the site structure on the users' visits. They found out that, although the data for the second period analyzed was collected shortly after the Web site change, the visits of the users were influenced by the new Web site structure as they usually visited together (in the same visit) pages of research teams from the same theme.

2.7.4 Other WUM Analysis Based on Our Preprocessing

In this section, we provide details about two other interesting WUM analysis based on our preprocessing. The first analysis conducted used the DS_2 presented in Table 2.14, while the last one contains the Web logs collected form the CS Department of the UFPE (Brazil). In Table 2.22, we provide a summary of DS_4, the initial size of the files and the size of the preprocessed files.

Dataset	**Web site(s)**	**Period**	**Size**	
			Before	**After**
DS_4	`www.cin.ufpe.br`	26.06.2002 – 26.06.2003	2,111MB	1,008MB

Table 2.22: Details of Dataset DS_4

The number of users, the user sessions and the visits identified for the first dataset is presented in Table 2.15, while the details concerning the second dataset are provided in Table 2.23.

Dataset	**Requests**	**Users**	**Sessions**	**Visits**
DS_4	6,787,675	517,645	517,655	1,266,303

Table 2.23: Results of Data Structuration for DS_4

2.7.4.1 Multi Dimensional Scaling Analysis on DS_2 [RLG05]

In [RLG05], the authors present a visualization method for WUM that enables the analyst to confront the syntactical organization of a Web site with the perception and understanding of this organization by the users of its Web site.

From the visits selected, according to the criteria listed in Table 2.24, the authors obtained generalized visits by considering only the first syntactic level of the URLs. The generalized visits were used to calculate the dissimilarities between the URL groups based on the syntactic topics of these Web sites. The URL groups were then projected

[5] The 190 research teams at INRIA are grouped in 5 research themes.

Ref.	**Objects**	**Criteria**	**Number**	**Analysis**
[RLG05]	Visits	Status = 2xx NbRequests ≥ 5 NbRequests ≤ 400	16,717	Multi Dimensional Scaling (MDS)

Table 2.24: Summary of DS_2 WUM Analysis

in two dimensions using the Multi Dimensional Scaling algorithm. This visualization was associated with a complementary representation of the minimum spanning tree induced by the dissimilarity matrix. The visualization highlights the relations between different URL groups (Figure 2.9). This method will allow the Web site editors to compare their expectations with the real usage of the Web site. For more details, the interested reader may refer to [RLG05].

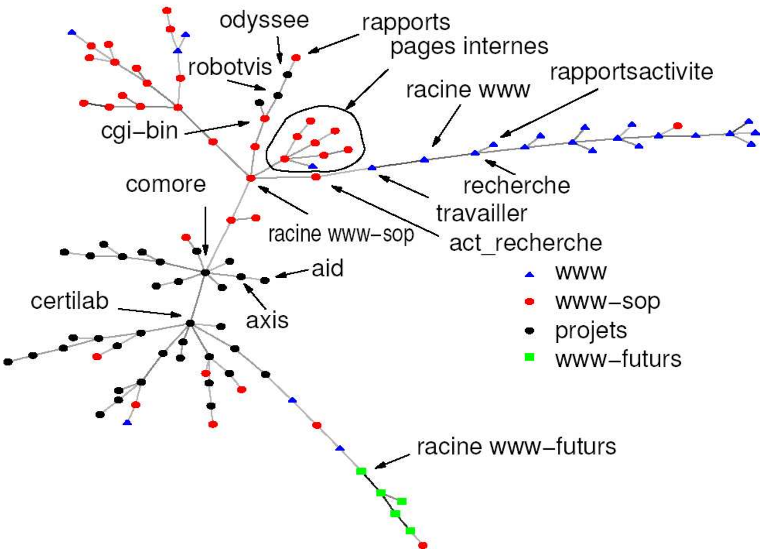


Figure 2.9: Minimum Spanning Tree [RLG05]

2.7.4.2 Statistical and Dynamic Clustering Analysis on DS_4 [Mar04]

The dataset analyzed in [Mar04][6] was collected over a period of one year from the `www.cin.ufpe.br`[7] Web server. The initial size of the log files was 2.1 GB and, after the preprocessing, it was reduced to 890MB. For this analysis, the author selected only the visits made during four months (corresponding to the first month of each quarter) and fulfilling the criteria enumerated in Table 2.25. Using this preprocessed data, in her

[6] This dataset was analyzed by a partner team from the University "Frederico II", Naples, Italy.

[7] Computer Science Department at the Federal University of Pernambuco from Recife (Brazil)

PhD thesis, Di Marco [Mar04] performed several multidimensional statistical analysis also followed by a number of clustering analysis.

Ref.	**Objects**	**Criteria**	**Number**	**Analysis**
[Mar04]	Visits from 07.2002, 10.2002 01.2003 and 04.2004	Status ≤ 304, $BS < 0.25$ Duration > 60s NbRequests > 10	63,171	PCA, MCA Dynamic Clustering

Table 2.25: Summary of DS_4 WUM Analysis

First, a Principal Component Analysis and a Multiple Correspondence Analysis were carried out. The variables used in these analysis were: `NbRequests`, `Duration` (called `TotalDuration`), `FileSize`, `AverageSize` and `AverageDuration`. The first three variables are calculated in the data summarization step of our methodology.

These analysis permitted to produce a first description of the log data and to identify homogenous classes of visitors (e.g. *users performing short visits with fast browsing*, *users performing short visits with slow browsing*, *users performing long visits with fast browsing* and *users performing long visits with slow browsing*).

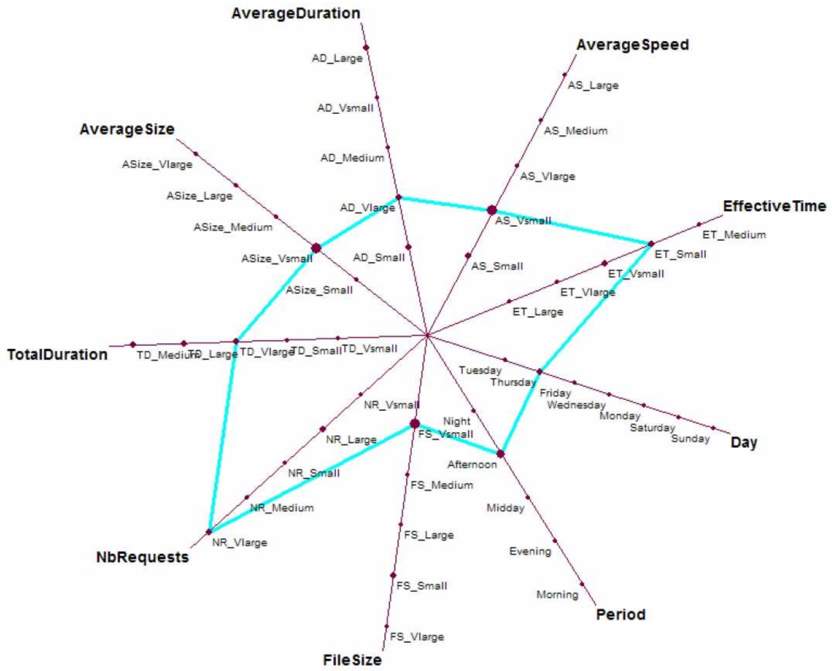


Figure 2.10: Star Diagram for a Prototype [Mar04]

The second analysis included techniques of Dynamic Clustering and Crossed Classification. The author used the S-CLUST module from the SODAS software[8]. S-CLUST is a non-hierarchical clustering technique that partitions a set of objects in a predefined number of clusters. The clusters are then represented by their prototypes using star diagrams (Figure 2.10). The results revealed correlations between the browsing typologies identified in the first analysis and the day of the week and the period of the day (the author split the day in 5 periods: “morning”, “noon”, “afternoon”, “evening” and “night”). For instance, she discovered that visits during the week day are generally done in the afternoons, they are either short or long but slow; during the week-end the visits are rather short, slow and are mostly done in the evenings and nights. One possible explanation could be the low speed of the dial-up connection that the visitor uses during the week-end to access the Web site.

2.8 Related Works in WUM Preprocessing

After illustrating the use of our WUM preprocessing methodology through different WUM processes, we will present in this section the main related works in this domain. In the recent years, we have seen much research on Web usage mining [BS00, CPY98, Coo00, FSS00, JK00, MDLN02, SK02, NK02, Wei04, Kum04, ESRR04, HBV04a, SBP04, VBYA04]. However, as described below, data preprocessing in WUM has received far less attention than it deserves.

2.8.1 Main Related Research Works in WUM Preprocessing

In this section, we describe and compare our methodology with the most significant works in this domain.

2.8.1.1 Cooley & al. [Coo00]

Cooley & al. [Coo00] presented methods for user identification, “sessionizing” (i.e. constructing or reconstructing sessions), page view identification, path completion, and episode identification. However, some of the heuristics proposed are not appropriate for larger, more complex Web sites.

For example, they propose to use the site topology in conjunction with the ECLF file for what they call the “user identification” (in our research, this is addressed under the term “user session”). The proposed heuristic aims to distinguish between users with the same IP address by checking every page requested in a chronological order. If a page requested is not referred by any previous page requested, then it belongs to a new user session. The drawback of this approach is that it considers only one way of navigating in a Web site, by following links. However, in order to change the current page, the users can, for instance, type the new URL in the address bar (most browsers have an autocompletion feature that facilitates this function) or they can select it from their bookmarks.

[8] SODAS software was developed by the ASSO project: `http://www.info.fundp.ac.be/asso/`.

Moreover, the sessionizing heuristic is complex as it defines the session boundaries according to five categories of Web pages (e.g. “head”, “media”, “navigation”, “look-up”, and “data entry”). Classifying Web site pages is not a simple task, due to their large number and because sometimes a page does not fit in just one of these five categories.

2.8.1.2 Berendt & al. [BS00, BMNS02]

In [BS00], the authors use the service-based conceptual hierarchy for modelling the query capabilities of an online catalog (SchulWeb). This Web site[9] offers browsing and searching services for schools worldwide. The authors wanted to evaluate the queries rather than the content of the Web pages generated, hence their service-oriented conceptual hierarchy. Concerning the preprocessing step, the authors propose three heuristics including one similar to the “Browsing Speed”. The other two heuristics refer to the number of requests without the referrer field and to the requests repeated for the same resource from the same host. However, we believe that the later one should be tested against other heuristics to validate its effectiveness.

In another work [BMNS02], the authors compared time-based and referrer-based heuristics for visit reconstruction. They found out that a heuristic’s appropriateness depends on the design of the Web site (i.e. whether the site is frame-based or frame-free) and on the length of the visits (the referrer-based heuristic performs better for shorter visits).

2.8.1.3 Huysmans & al. [HBV04b]

In [HBV04b], the authors analyzed the Web log data from an online wine shop (e-shop). They developed a tool for preprocessing (in Delphi) and implementing some of the operations already described in this chapter and shown in Table 2.26. They mentioned that the preprocessing of this data, and especially the images removing, reduced its initial size by a factor of 10 (which is close to the figures we presented in the Section 2.7). This was obviously due to the type of the Web site analyzed, as e-shops and e-commerce Web sites, in general, have many images in order to display the products sold. Another interesting remark made by the authors is that by using a classical log analyzer program (*Analog* in their case), the size of the log file was reduced only by 15%, which shows again that appropriate tools are essential for an effective preprocessing step.

2.8.1.4 Marquardt & al. [MBR04]

In [MBR04], Marquardt et al. addressed the application of WUM in the e-learning area with a focus on the preprocessing phase. In this context, they redefined the notion of visit from the e-learning point of view. In their approach, a *learning session* (*LS*), visit in our case, can span over several days if this period corresponds to a given learning period. A learning session may also correspond to the set of accesses made to accomplish a given task. The authors also identified episodes similar to [Coo00], by

[9] http://www.schulweb.de

classifying Web pages into three page types (auxiliary, content and resource) based on the existing knowledge about the Web site.

2.8.1.5 Bonchi & al. [BGG$^+$01]

Bonchi et al. [BGG$^+$01] also developed a data warehouse for storing Web log files. Unlike our relational model, their model does not contain structured information about the usage (sessions, visits, etc.), users or aggregated variables. The objectives of their work was Web caching, therefore, the data warehouse, implemented in Microsoft SQL Server, was populated with Web logs from several Web servers. For Web caching applications, all the requests present in the Web logs are important and have to be stored, thus, the preprocessing step is different from other WUM tools because almost all requests are kept. Regarding the user and the session identification, the authors only use the notion of "user", identified using the IP heuristic.

2.8.1.6 Other WUM Research Works

As shown, the preprocessing step is important and should be present in all WUM analysis. Therefore, we compared our preprocessing methodology with the preprocessing described in other general WUM research works [CPY98, FSS00, JK00, SK02]. The results of this comparison are provided in Table 2.26.

2.8.2 Comparison Criteria for WUM Preprocessing

Table 2.26 compares our method to the above mentioned WUM research works. The comparison table focuses only on the data preprocessing step and the table shows how different preprocessing features where implemented in the main related works. We enclose below a brief description of these features grouped by the preprocessing step:

- **Data Source** – represents the data used as input in the WUM preprocessing.
- **Data Preparation** – includes the operations carried out when several log files are being preprocessed and when the analyst intends to "de-personalize" the log file through annonymization.
- **Data Cleaning** – contains the main filtering operations used for eliminating: images, Web robots' requests and requests for other non-analyzed resources. The removal of Web robots can be done with one of the heuristics presented in Section 2.4.2: detecting requests for **Robots.txt** (RT), using lists of **known user agents** (UA) or the **Browsing Speed** (BS).
- **Data Structuration** – includes all the processes used for grouping the requests (in sessions, visits, etc.).
- **Data Reduction** – represents the final step of the preprocessing and contains all the operations that are designed to reduce the size of the data, like data generalization and data summarization.

Related Work	**Data Source**			**Data Preparation**		**Data Cleaning**				**Data Structuration**				**Data Reduction**	
	Site		Multiple Web sites	Joining	Anonymization	Removing images	Removing WR			User	Session	Visit	Episode	Generalization	Summarization
	Map	Semantic					RT	UA	BS						
[Coo00]	√				√	√	√	√		Login	(IP, Agent)	H_{Visit}, H_{Ref}	Page type		
[BS00]						√			√		IP	H_{Visit}			
[HBV04b]						√	√				(IP, Agent)	H_{Page}	MF		
[MBR04]						√				Login	Login	H_{Ref}, LS	Page type		
[BGG$^+$01]			√	√						IP					
[CPY98]	√											MF			
[FSS00]						√					IP	H_{Visit}		√	
[JK00]						√					IP				
[SK02]	√				√						SessionID	H_{Page}			
Our Method	(√)	√	√	√	√	√	√	√	√	Login	(Login, IP, Agent)	H_{Page}	Semantic Topics	√	√

Table 2.26: Comparative Analysis of Our Work with the Related Works

In conclusion, our methodology is more complete because:

- It offers the possibility of analyzing jointly multiple Web server logs;
- It can define a hierarchy of semantic page topics that can also serve to generalize pages and identify episodes;
- It employs effective heuristics for detecting and eliminating Web robot requests;
- It proposes a complete relational database model for storing the structured information about the Web site, its usage and its users;
- It uses an advanced preprocessing step for enriching the structured information by calculating different statistical measures for the objects stored in the database and it also facilitates the addition of new user-defined measures (that will be used as parameters in the data mining step).

2.9 Discussion and Perspectives

Our experiments presented in Section 2.7 showed the importance of the data preprocessing step and the effectiveness of our methodology, by reducing not only the size of the log file but also by increasing the quality of the data available through the new data structures that we obtained. The relational database model that we proposed in Section 2.6 can be easily extended by adding new parameters depending on the objects analyzed (as columns in the existing tables) or by defining new objects (as new tables). These parameters will depend on the Web site(s) analyzed and also on the purposes of the WUM analysis.

Although the preprocessing methodology presented allows us to reassemble most of the initial visits, the process itself does not fully guarantee that we identify correctly all the transactions (i.e. user session, visits and episodes). This can be due to the poor quality of the initial log file as well as to other factors involved in the log collection process (e.g. different browsers, Web servers, cache servers, proxies, etc.). These misidentification errors will affect the data mining, resulting in erroneous Web access patterns. Therefore, we need a solid procedure that guarantees the quality and the accuracy of the data obtained at the end of data preprocessing.

A first solution would be to request the user to register, in the case of the Web site belonging to the category of Web sites commonly employing this method (for instance e-commerce Web sites). If the nature of the Web site does not allow this, other alternative solutions could be used, like cookies or a session ID in conjunction with dynamic pages. The first alternative has the advantage that a returning user is detected. However, privacy would be an issue and cookies can be blocked by users. With the second alternative (URLs with session IDs), each time that a new user connects to the Web site, a session ID would be generated. This ID will be dynamically included in all the links from the pages he/she requests. However, this will not work when the user will type or select from the bookmarks the address of a new page from the Web site. Moreover, this will function only for identifying visits done to the Web site during one session. A user that returns to the Web site will receive a new session ID that cannot be linked to the previous ones.

A solution that would allow to increase the quality of the data collected (i.e. the log file), consists in the use of a Java applet or a Java script that would automatically send a message back to the Web server when the page is fully loaded. This event would allow the analysts to compute the real display time by making the difference between the request time of the following page and the load time of the current page.

In Section 2.5.3, we affirmed that having the site structure (via the site map) is vital for the page view identification. However, a Web site's current structure might not coincide with its structure at the time when the logs were collected. Therefore, we need a mechanism to efficiently manage different versions of the site structure. For example, we could use XGMML [PK01] source files with the Concurrent Versions System tool.

The Web site map could also be used for path completition [Coo00]. This is useful when not all of the user requests are present in the log file due to cache and proxy servers.

More details about our perspectives for the WUM preprocessing and the WUM process in general are given in the Chapter 5.

Chapter 3

Methods for Sequential Pattern Extraction in WUM

The Web Usage Mining techniques provide knowledge about the users' behavior on a Web site, knowledge expressed through the relations (patterns) hidden in the log files. Among the techniques available [CMS99, SFW99, MPC99, MDLN02, ESRR04, XD01, FSS00, SBP04], the sequential pattern mining is particularly well adapted to the log study due to the sequential nature of the Web site users activity. The extraction of sequential patterns from a log file is assumed to provide this kind of pattern for a given Web site:

Example 1 *On INRIA's Web Site, 10% of users visited consecutively the following pages: the homepage, the positions available for engineers and technicians (ET), the ET's missions, the competitive selection for ET and the FAQ about these positions.*

This pattern is just an assumption, because in reality, an analyst has to manage several problems, such as the cache, the proxy servers, the great diversity of pages on the Web site, the research engine allowing a direct navigation to a Web page etc. Moreover, the proportion of the JOB OFFERS part of INRIA's Web site compared to the entire Web site is approximately 1.5%, and the proportion (given by the pattern's support) of the users following this browsing pattern through the JOB OFFERS part of the site is even less than that.

If the caching problems can be solved [Coo00], the low support issue needs a more in depth study. In order to illustrate this, let us consider the sequential pattern we are supposed to obtain (Example 1). Because of the small size of the "JOB OFFER" part of INRIA's Web site, the number of visits that contain these pages (in the same order) represent less than 0.16% of all the visits. In another example, the users that made requests to the teaching pages of J. Smith represent approximately 0.7% of the total number of users (see Section 3.7.3.1 for more details about these patterns). A WUM study on such a Web site, has to manage this particularly low support in order to provide satisfying results.

3.1 Motivations

Our goal is to discover patterns revealing behaviors with a very low support. For a complex Web site with numerous users, these patterns will correspond to groups of minorities that are, for example, interested in specific parts of a Web site. For instance, let us consider two of INRIA's Web sites: the main site – `www.inria.fr` and the site of INRIA Sophia Antipolis – `www-sop.inria.fr`. These Web sites can be represented as shown in the Figure 3.1 below, for the pages containing information about the "RESEARCH", "TEACHING", "JOBS" topics.

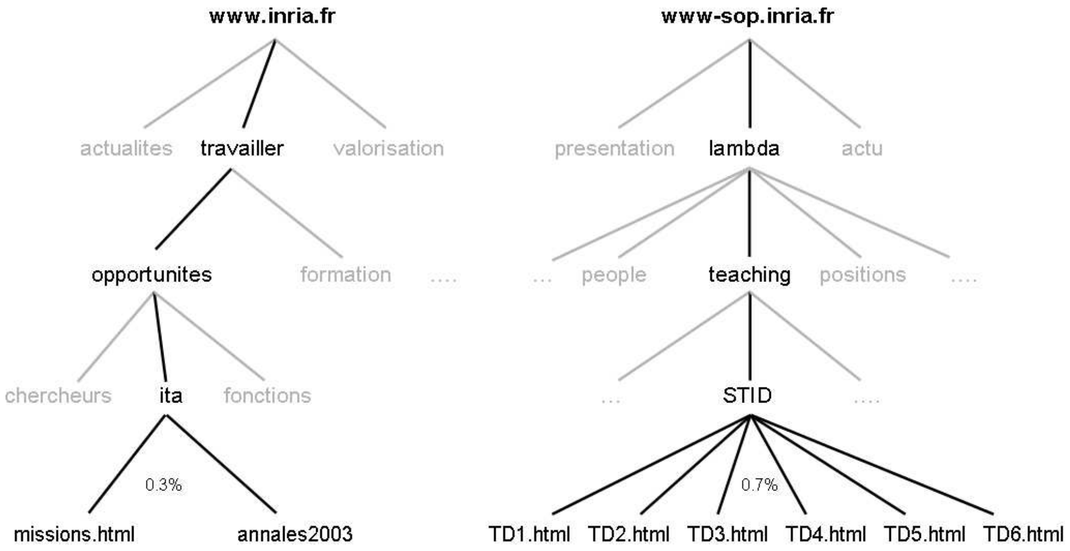


Figure 3.1: Sections from Two of INRIA's Web Sites

From a log analysis conducted on these two Web sites, we can provide the following lessons:

- The interesting behaviors are contained in a specific part of the log. For example, in Figure 3.1, the part corresponding to the teaching activities of J. Smith (STID) will be requested by **0.7%** of the users recorded in the log while the users interested in the job opportunities for the engineers and technicians will represent **0.3%** of all requests on the site.

- However, the sequential patterns coming from such a log can be disappointing. In fact, their relevance is weak and sometimes they can be obvious and not very useful (e.g. "1% of users arrive at the homepage of `www-sop.inria.fr` and then go to the presentation page").

- In order to obtain interesting patterns on a log file, we have to specify a very low support for the sequential pattern mining algorithm.

In our research framework, we exclude methods with constraints and sampling methods. Without denying the efficiency of the methods that use **constraints**, we argue that these techniques do not allow us to find all the patterns (which are still to be discovered, thus unknown from the user due to the constraints used). For instance, in order to discover the pattern from Example 1, the analyst must search for patterns that include at least one employment page belonging to that pattern. Regarding the **sampling** techniques, we consider that the representativeness we are working with is so weak that the size of a sample will be almost the same as that of a log.

For a classical sequential pattern mining algorithm, two problems can appear when specifying a very low support:

- The response time will be very long (in most cases, the results will not even be obtained due to the exponential complexity of the process);
- The amount of the frequent patterns generated by this process (in case the process terminates) would be huge (i.e. several thousands).

Nevertheless, the behaviors we want to discover have a very low support. They correspond to minorities, but we aim at discovering these kind of patterns since we consider that they are highly relevant. For instance, among these behaviors, we could notice hacking activities or students' sessions on specific teaching pages. Our goal is to provide patterns, revealing behaviors such as:

- **0.02**% of the users have a session similar to **hacking activities**. Among them, 90% followed a typical hacking session (i.e. our system administrators confirmed that the requests discovered correspond to those implemented by known hacking utilities, used by script kiddies);
- **0.7**% of the users have a session related to the **teaching pages** of J. Smith. Among them, 15% requested consecutively the 6 pages of his course on data mining.

The very weak support of these patterns is mainly due to the great diversity of the behaviors on the logs analyzed and also to the large number of URLs contained in that site.

To overcome these issues, we propose in this chapter three new approaches for extracting sequential patterns with low support from a structured log file: the Sequential Approach, the Iterative Approach and the Hierarchical Approach. All three approaches are based on a divisive principle, i.e. as the log file cannot be mined directly with an sequential pattern mining algorithm, the file is divided first into sub-logs and the sequential pattern mining algorithm is applied afterwards on them.

This chapter has the following structure: in Section 3.2, we define the problem of extracting sequential patterns and give the challenges of the sequential pattern mining algorithms when applied to WUM. Our general methodology and the three approaches we proposed based on this methodology are described in Section 3.3. The Cluster & Discover method, implementing our first approach, the Sequential Approach, is presented in Section 3.4. The Divide & Discover method, implementing our second approach, the Iterative Approach, is presented in Section 3.5. The Hierarchical Discover method, implementing our third approach, the Hierarchical Approach, is presented in Section 3.4 In Section 3.7 we give the results of the experiments we conducted, while the main related works addressing similar problems are presented in Section 3.8. Finally, our conclusions and perspectives are enounced in Section 3.9.

3.2 Introduction to Sequential Pattern Mining

The *sequential pattern mining* is the mining of frequently occurring patterns related to the time or other sequences [HK01]. In *E*xample 1, we described a frequent sequential pattern related to other sequences in the structured log file (i.e. 10% of all the sequences).

Using sequential pattern mining, one can easily identify the paths that users frequently follow on a Web site. Therefore, they are widely used in the WUM field, as mentioned in Section 1.2.2. In this section, we give a brief description of the sequential pattern mining problem, then we explain the issues arising when applying sequential pattern mining techniques in WUM and, finally, give some limitations of the actual algorithms.

3.2.1 Definitions

The sequential pattern mining (SPM) is based on association rule mining, and we present hereby the definitions for both terms, as given by [AIS93] and [SA96]. In [AIS93], an **association rule** is defined as follows:

Definition 1 Let $I = \{i_1, i_2, ..., i_m\}$ be a set of m literals (*items*). Let $D = \{t_1, t_2, ...t_n\}$ be a set of n transactions. Associated with each transaction is a unique identifier called its TID and an *itemset* I. I is a *k-itemset* when k is the number of items in I.
We say that a transaction T *contains* X, a set of some items in I, if $X \subseteq T$.
The *support* of an itemset I is the fraction of transactions in D containing I: $supp(I) = \|\{t \in D \mid I \subseteq t\}\| / \|\{t \in D\}\|$.
An *association rule* is an implication of the form $I_1 \Rightarrow I_2$, where I_1, $I_2 \subset I$ and $I_1 \cap I_2 = \emptyset$. The rule $I_1 \Rightarrow I_2$ holds in the transaction set D with confidence c if $c\%$ of transactions in D that contain I_1 also contain I_2. The rule $r : \; I_1 \Rightarrow I_2$ has *support* s in the transaction set D if $s\%$ of transactions in D contain $I_1 \cup I_2$ (i.e. $supp(r) = supp(I_1 \cup I_2)$).

Given two parameters specified by the user, *minsupp* and *minconfidence*, the problem of association rule mining in a database D aims at providing the set of frequent itemsets in D, i.e. all the itemsets having a support greater than or equal to *minsupp*. Association rules with a confidence greater than *minconfidence* are thus generated.

The definition for the association rule does not take into consideration the time, contrary to the **sequential pattern** which is defined in [SA96] as follows:

Definition 2 A *sequence* is an ordered list of itemsets denoted by $< s_1 s_2 \dots s_n >$ where s_j is an itemset. The *data-sequence* of a customer c is the sequence in D corresponding to customer c. A sequence $< a_1 a_2 \dots a_n >$ is a *subsequence* of another sequence $< b_1 b_2 \dots b_m >$ if there exist integers $i_1 < i_2 < \dots < i_n$ such that $a_1 \subseteq b_{i_1}, a_2 \subseteq b_{i_2}, \dots, a_n \subseteq b_{i_n}$.

Example 2 *Let C be a client and S*$=<$ (3) (4 5) (8) $>$ *be the purchases of that client. S can be read as "C bought the item 3, then the items 4 and 5 at the same moment (i.e. in the same transaction) and, finally, C bought the item 8".*

Definition 3 The *support* for a sequence s, also called $supp(s)$, is defined as the fraction of the total data-sequences that contain s. If $supp(s) \geq minsupp$, with a minimum support value *minsupp* given by the user, s is considered as a *frequent* sequential pattern.

3.2.2 Log Files Analysis with Sequential Patterns

The format of the (access) log file is not suitable for directly applying sequential pattern mining algorithms on them. This is due to the fact that the sequential pattern mining techniques were designed at first to analyze the market basket transactions. Therefore, the log data needs to be transformed into a *(client, date, item)* format.

The general process of Web Usage Mining is similar to the principle proposed in [FPSS96]. It relies on three main steps. First, starting from a raw data file, a preprocessing step is needed to clean the "useless" information as presented in Chapter 2. The second step starts from this preprocessed data and applies data mining algorithms to find frequent itemsets or frequent sequential patterns. Finally, the third step aims at helping the user to analyze the results by providing a visualization of the result.

Raw data is collected in log files by Web servers. Each input in the log file illustrates a request from a client machine to the server (*http daemon*). Access log file formats can differ, depending on the system hosting the Web site. We illustrate our approaches with an access log file in the ECLF format (an extended version of the Common Log Format (CLF) given by CERN and NCSA [NCS95]). A log entry contains 9 fields, separated by spaces:

host user authuser [date:time] "request" status bytes "referrer" "user agent"

Definition 4 Let *Log* be a set of server access log entries. An entry g, $g \in Log$, is a tuple $g =< ip_g, ([l_1^g.URL, l_1^g.time] \ ... \ [l_m^g.URL, l_m^g.time]) >$ such that for $1 \leq k \leq m$, $l_k^g.URL$ is the item requested by the user ip_g at time $l_k^g.time$, for all $1 \leq j < k$, $l_k^g.time > l_j^g.time$.

User \ Item (position)	1	2	3	4	5
1	10	30	40	20	30
2	10	30	20	60	50
3	10	70	30	20	30

Table 3.1: File Obtained after the Preprocessing Step, from the DB

The preprocessing applied on the access log files and described in Chapter 2 is used for cleaning the log files and grouping the requests in sessions[1]. At the end of this process, the preprocessed data is stored in a relational database. From this database, it is straightforward to extract a file in the format *(client, date, item)*, containing the sessions corresponding to all the users we want to analyze. The Table 3.1 gives an example of a file extracted from a database containing a preprocessed log for 3 users. For each user, we have an ordered list with the URLs requested. For instance, the user 2 requested the URL with the ID *"60"* in the forth requests.

Thus, our goal is, according to Definition 3 and by means of an sequential pattern mining step, to find in this file the sequential patterns that can be considered as frequent. The result may be, for instance `< ( 10 ) ( 30 ) ( 20 ) ( 30 ) >` (using the file illustrated in Table 3.1 and a minimum support of 66%). Such a result, once mapped back into the URLs, allows the discovery of a frequent behavior, common to n users (with n calculated as the support of the patterns times the total number of users) and also gives the sequence of the events (i.e. clicks) composing that behavior.

3.2.3 Challenges for Sequential Pattern Mining Algorithms

As mentioned in Chapter 1, the increase in the Internet usage determined the increase of the Web logs. Moreover, Web sites are becoming larger as more Web pages are being added than suppressed. Web site owners prefer to keep the old Web pages because storing information is relatively cheap nowadays and also because users bookmark Web pages and they expect to find them always online.

Sequential pattern mining algorithms are usually based on the well known Apriori principle of the *candidate generation* [AIS93]. In this type of algorithms, at a given step k, candidate patterns of length $k+1$ are generated from current patterns of length k and then tested against all the sequences in the database. This strategy proved to be inefficient for a large number of sequences or a large number of items as the latter will imply a large number of candidates being generated. In the case of Web logs, we have a high number of both sequences (i.e. sessions) and items (i.e. Web pages). Therefore, applying standard algorithms for extracting the sequential patterns from Web logs will be possible only for higher supports (the number of candidates generated will rapidly decrease). But, this will lead to few or uninteresting results (e.g. short patterns or

[1] In this chapter we use the term *session* to designate the objects analyzed, representing the user sessions, the visits or the episodes.

patterns involving the main pages of the Web site). For example, one may discover that *"20% of the users visited the home page and then the research teams page"*.

To obtain more interesting sequential patterns, i.e. longer or involving pages with a higher *depth* or syntactic level, we need to be able to extract the sequential patterns with a very low support. The existing sequential pattern mining algorithms are not capable of doing this because of the exponential number of candidate patterns being generated at each step. For example, we can assume that for a very low support s_{vl}, the algorithm generates n_1 patterns of length 1. The number of candidates of length 2 is n_1^2 and these candidates need to be tested against all the sequences from the database.

To face these challenges, we propose to use a general divisive sequential pattern mining methodology and we designed three approaches for implementing it.

3.3 Three Divisive Sequential Pattern Mining Approaches

In this section, we describe the general principle of our sequential pattern mining methodology and present the three approaches we propose based on this methodology.

3.3.1 General Principle

The governing **principle** of our methodology is to divide the initial problem of extracting sequential patterns from a large preprocessed log file, in n similar sub-problems, one for every sub-log generated from the initial file. The log file is initially split into sub-logs. Then, an sequential pattern mining algorithm may be applied to extract sequential patterns on a sub-log or this one may be further split, depending on the approach chosen.

The three **approaches** that we propose here are:

1. The **Sequential Approach**, consisting first in a clustering step applied on the sessions, followed by an sequential pattern mining step on each of the clusters generated.

2. The **Iterative Approach**, composed of an sequential pattern mining step applied on all the sessions with the lowest possible support, followed by a clustering step executed on the output of the sequential pattern mining step (i.e. the sequential patterns). The sessions are then classified based on the output of the clustering step and the entire process is used iteratively on the remaining unclassified sessions.

3. The **Hierarchical Approach**, including an sequential pattern mining step applied on the sessions with a fixed hierarchical level[2] for the Web pages. The log is

[2] The syntactic/semantic topics are organized as an hierarchy, thus, the hierarchical level n of a page

split using the results of the sequential pattern mining step and, thus, this process is repeated for some large sub-logs. The hierarchical level of the Web pages will be increased in the sessions belonging to these sub-logs.

3.3.2 Sequential Approach

The Sequential Approach is based on the assumption that, with a proper clustering method, similar sessions can be grouped together into the same cluster. Then, mining the sessions from this cluster with an appropriate sequential pattern mining algorithm will permit the discovery of interesting patterns with very low support. Finally, the support of these patterns is recalculated with respect to the initial file. The representativeness of the sequential patterns in the cluster is significantly higher when the cluster contains similar sessions. Therefore, an sequential pattern mining algorithm will be capable of extracting the common patterns for these sessions in the cluster. The schema in the Figure 3.2 represents this approach.

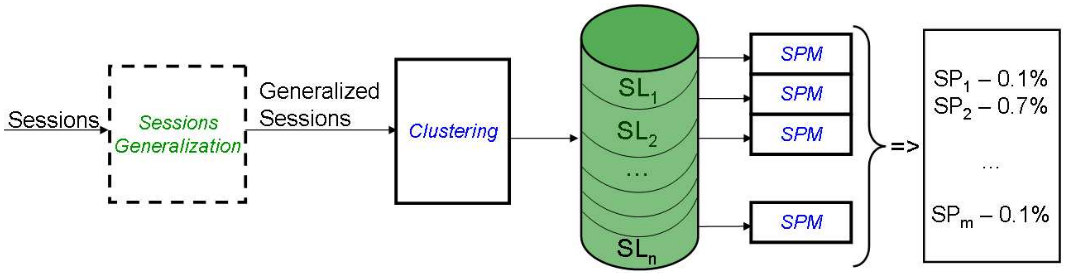


Figure 3.2: The Sequential Approach

A method that will instantiate the Sequential Approach must have the following 3-three steps general principle:

Step 1. Clustering: By means of a clustering technique, partition the preprocessed log file (structured in sessions) in several classes (sub-logs).

Step 2. SPM: Extract the frequent sequential patterns from each class by applying a sequential pattern mining algorithm. To specify the minimum support for the sequential pattern mining algorithm, several strategies, as described below, can be used.

Step 3. Support: Extend the support of each pattern discovered with respect to the initial log.

This Sequential Approach is easily instantiated by choosing the proper clustering and

is the generalized page at the syntactic/semantic level n.

sequential pattern mining methods. The clustering method will, eventually, include a preprocessing step, where the sessions are transformed into numerical vectors (the "sessions generalization" step). The key requirement for the clustering step is that sessions supporting the same patterns should be clustered together into the same cluster. For this, the clustering technique which will be employed, could use a similarity function based on the number of common pages between sessions and, eventually, the order of these pages.

For establishing the support of the SPM step, we propose the use of a value s as the *global support*, chosen at the beginning of the whole process. The value for the support in a cluster C_k, called *local support in* C_k, will be calculated as:

$$s(C_k) = s\frac{|S|}{|C_k|},$$

where $|S|$ is the number of the sessions included in the initial file and $|C_k|$ is the number of the sessions belonging to the cluster C_k.

A second possible solution would be to let the user manually choose a support value for each of the clusters discovered.

Once the sequential patterns are extracted from all the clusters, they need to be grouped and pruned, as the same pattern can be extracted from two different clusters (especially, short patterns). Also, the support for each pattern will be recalculated in the initial file. This is a linear process and therefore it is not time consuming.

The final patterns may be presented to the analyst grouped by cluster (as a cluster represents similar interests) and/or ordered by the global support value.

We instantiated the Sequential Approach by choosing a neural clustering [BT02] and the PSP algorithm [MPC99] to obtain the Cluster & Discover (C&D) method. This method is described in detail in Section 3.4.

3.3.3 Iterative Approach

The Iterative Approach is based on the following observation: we noticed that for sequential patterns extracted using different levels of support, generally, for higher support values shorter patterns were obtained. This common sense property is based in fact on the Apriori principle which states that if a sequence S_i is included in another sequence S_j, then all the subsequences of S_i will be included in S_j. Therefore, if S_i is a frequent sequential pattern, then all the subsequences of S_i, shorter than S_i will also be frequent sequential patterns.

Consider now a pattern P, with a very low support such that a direct sequential mining process cannot extract P from the sessions' file. However, when mining this file with higher support values, some of the sub-patterns from P will be obtained as their support is greater than the support of P (Apriori principle). The sequences supporting these sub-patterns, if grouped together, can lead the analyst to the discovery of P.

Based on this assumption, we designed the Iterative Approach using the following 4-step principle, as presented in Figure 3.3:

Step 1. Extract sequential patterns from the structured log file. The support used within this step should be as low as possible in order to extract a maximum number of patterns in this first step.

Step 2. Cluster these sequential patterns using an appropriate clustering method (e.g. neural clustering). The greater the number of patterns available, the more accurate will the division process be.

Step 3. Divide the structured log file according to the clusters obtained in Step 2. Each sub-log contains sessions from the original log, corresponding to at least one pattern from its cluster. A special sub-log needs to be created to collect the sessions from the original sub-log which do not correspond to any of the clusters obtained in the previous step.

Step 4. Depending on the size of the sub-logs obtained in Step 3, re-apply the whole process (described above in Step 1, 2 and 3) iteratively for the sub-logs requiring further analysis.

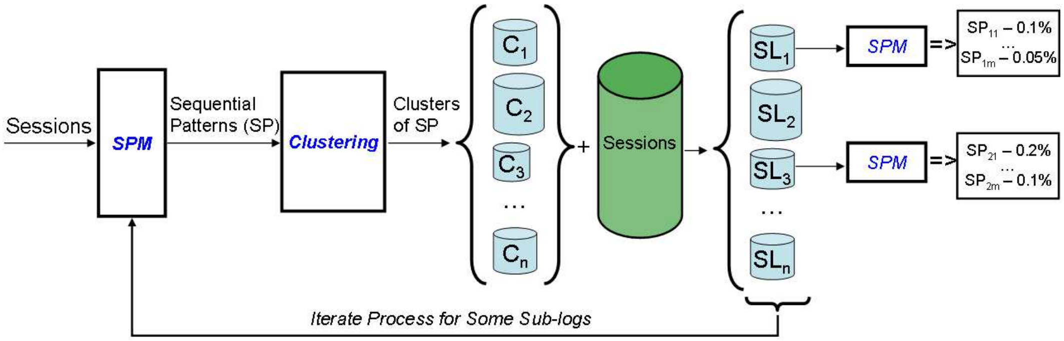


Figure 3.3: The Iterative Approach

For the Iterative Approach, the clustering step is applied on the sequential patterns and not directly on the sessions as in the Sequential Approach. This requests significantly less objects to be clustered and, therefore, a wider range of clustering techniques can be used.

On the other hand, in the Iterative Approach, the SPM step is applied in Step 1 on the entire sessions' file and it will take longer as the support decreases. However, once the file is split, the sequential pattern mining algorithm will be applied on smaller and more homogenous clusters, thus making the process more efficient and capable of extracting patterns with very low support.

In this approach, the analyst can drive the entire data mining process by choosing on which clusters to continue the iterative process and the support value to be used for these clusters in the second iteration.

Another option for dealing with the support issue in this approach would be to implement an automated version in which the last cluster (containing the sessions that do not belong to any clustered pattern) is further split. In this case, the support value would be exponentially decreased at each iteration level and the process would be repeated until the size of the cluster goes below a specified threshold.

We instantiated the first option of this approach (i.e. the analyst-driven approach) in the method called Divide & Discover (D&D), described in [MTT04b]. A detailed description of this method is given in Section 3.5.

3.3.4 Hierarchical Approach

The Hierarchical Approach (see Figure 3.4) is also an iterative approach, but at each iteration, the level of detail used for the sessions' generalization is increased. The sessions are generalized before the SPM step and the Web pages are replaced with more general syntactic or semantic topics of a specified level. For instance, the page `www-sop.inria.fr/lambda/teaching/STID/ex1.html` can be replaced with `www-sop.inria.fr/lambda/teaching/` which represents its second level syntactic topic. If a semantic hierarchy is available for the Web site analyzed, the generalization can be done based only on this hierarchy.

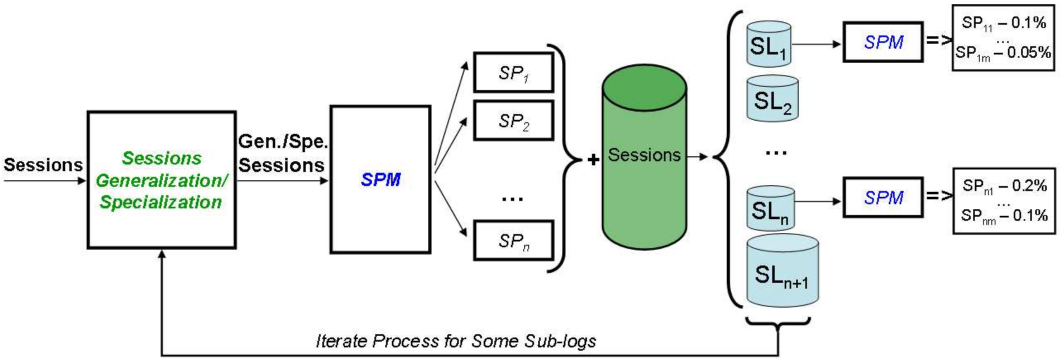


Figure 3.4: The Hierarchical Approach

The sequential pattern mining algorithm is applied on these generalized sessions. The number of items (i.e. generalized Web pages) is significantly reduced in this case compared with the initial log file as we keep only the top level topics from the syntactic (or semantic) hierarchy. This allows us, on one hand, to find more representative sequential patterns in the SPM step and, on the other hand, to extract patterns with lower support values as the number of different items is smaller.

The clustering step is not included in this approach. Once the sequential patterns are obtained, the log file is split based on these results and one sub-log is created for each of the patterns discovered. All sessions belonging to this sub-log support the pattern, i.e. they include a sequence of Web pages which once generalized would represent the sequential pattern. As not all the sessions support one of the patterns discovered, a new sub-log, S_{n+1} will be created and all these "original" sessions will be placed there.

The entire process may be repeated on some of the sub-logs or only on S_{n+1}, if required. Another possibility is to mine the sub-logs using an sequential pattern mining algorithm. When iterating the process, the hierarchical level of the generalized sessions is increased, we say in this case that the generalized sessions are **specialized**. The increase in the level of detail at which we consider the (sub-)log file is similar with "zooming" on this (sub-)log file.

The Hierarchical Approach is instantiated in the Hierarchical Discovery method presented in Section 3.6.

3.3.5 Applying Our General Divisive Methodology

To illustrate and support our divisive methodology and sequential pattern mining approaches, we developed three different methods: "Cluster and Discover" (C&D), "Divide and Discover" (D&D) and "Hierarchical Discovery" (H&D). These methods represent instantiations of the three approaches using specific algorithms for clustering and sequential pattern mining. The summary of these three methods is presented in Table 3.2.

METHOD		**C&D**	**D&D**	**H&D**
Clustering	Input	Sessions	Patterns	-
	Clusters	Disjunctive	Non-disjunctive	-
Pattern Extraction	Algorithm	PSP	PSP	Apriori-GST
	Output	SPs	SPs	Contiguous SPs

Table 3.2: Summary of the Three Methods Implemented

In Figure 3.5, we represent the three methods and compare them against a classic sequential pattern mining, Apriori-like, algorithm.

For the clustering step that we implemented in the C&D and D&D methods, we use a neural cluster algorithm developed in the AxIS research team and described in Section 3.4.2. This algorithm is applied on sessions in the C&D method and on sequential patterns in the D&D method. The result of the log division is non-disjunctive for the D&D algorithm as one session can support several patterns (i.e. thus it will belong to several sub-logs).

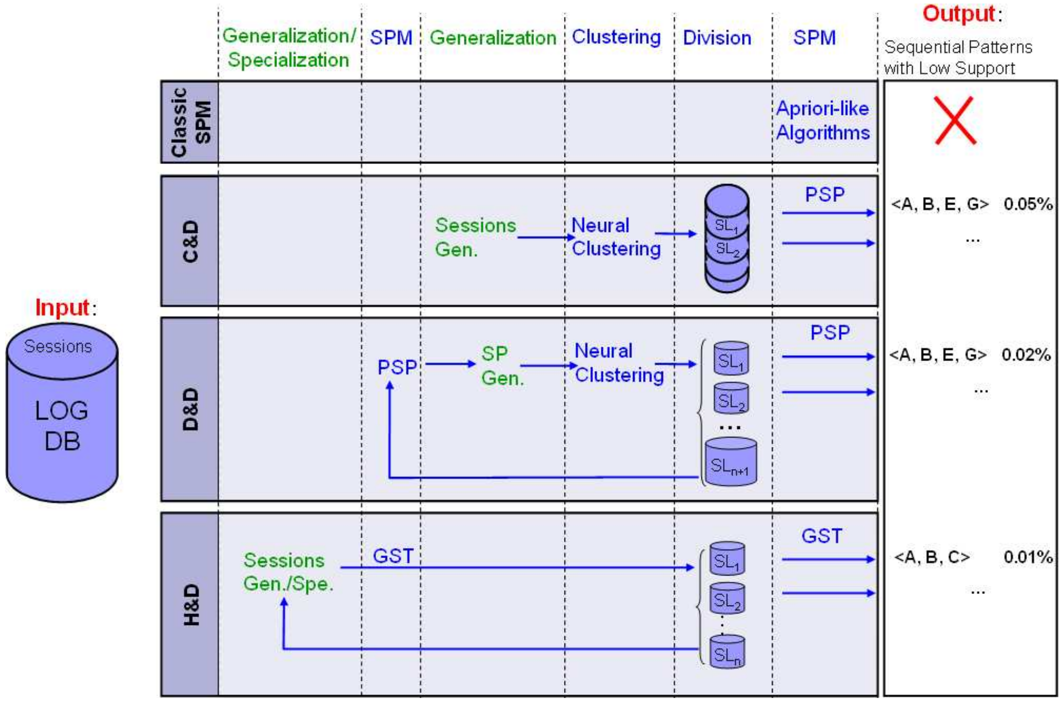


Figure 3.5: Comparison Between Classic Sequential Pattern Mining and the Three Methods Implemented

To apply the clustering algorithms on the sessions or sequential patterns, we need to summarize them as input vectors. For this purpose, we developed a method based on the semantic topic of the page visited as presented in Section 3.4.1.

For the SPM step, we use two Apriori-like algorithms: PSP [MPC99] in the C&D and D&D methods, and respectively Apriori-GST (described in Section 3.6.1) in the H&D method. The difference between the two algorithms is that the Apriori-GST extracts contiguous sequential patterns using a generalized suffix tree (GST) index, while the PSP extracts sequential patterns as described in Section 3.2.

These three methods will be described in detail in the next sections.

3.4 Sequential Approach Illustration: Cluster & Discover Method

We instantiated our Sequential Approach in the Cluster & Discover (C&D) method. Our objective is to discover classes of users (grouped according to their behavior) and to analyze their sessions by extracting the frequent sequential patterns. The method has three main steps as represented in Figure 3.6. These steps are implemented with the following algorithms:

1. The clustering step is implemented using a **neural clustering** algorithm applied on the users' sessions. We use the same neural clustering algorithm, as described in [BT02], that we also use in the D&D method. However, here, the algorithm is applied directly on the initial log file. We employed a threshold on the size of the clusters for determining the representative clusters. The rest of the clusters, considered as atypical clusters, are joined in a single class (sub-log) called C_A. At the end of the clustering step, the log file is split into distinct and disjunctive classes that regroup similar behaviors.

2. The sequential pattern mining algorithm that we use for extracting sequential patterns from each class is the **PSP algorithm**, introduced in [MPC99]. We use a unique pre-set support for all the classes, to minimize the user interaction with the system.

3. In the last step, we compute the global support for each pattern p_i discovered in the previous step. This is done by searching in the initial log the set $\{S_j\}$ of all the sessions that support (match) p_i. The global support is then calculated by dividing the number of these sessions by the total number of sessions.

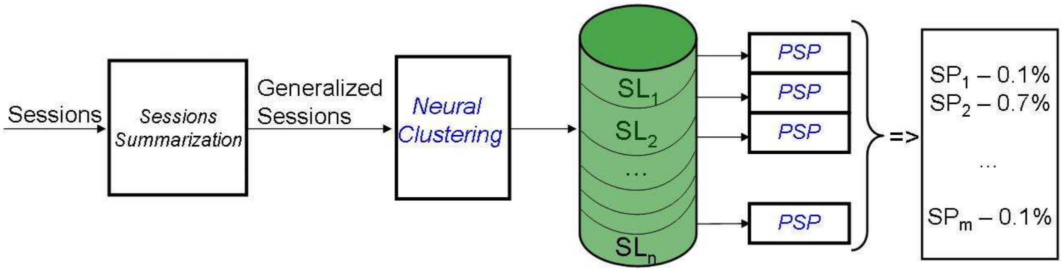


Figure 3.6: The C&D Method

In this method, we need a session generalization step before the clustering step in order to transform the sessions into a suitable format for the neural clustering algorithm (i.e. a fixed-length numeric vector). In the following section, we describe this preliminary step.

3.4.1 Sessions Summarization Using Semantic Topics

A session can also be considered a sequence of Web pages. For the neural clustering algorithm, we need to transform this session into a vector of positive integers, called the *description vector*. Therefore, we use four attributes to characterize a session. These attributes are based on a *generalization* of its Web pages.

The first two attributes consider the *multi-site* aspect of the sessions while the last two attributes consider their *first-level semantic topic* aspect. As mentioned in Section 2.5.5, a semantic topic groups similar syntactic topics. For instance, syntactic topics of level one in the `www-sop.inria.fr` Web site such as *axis, everest, lemme* are grouped together under the *projets* semantic topic, *i.e.* *"research teams"*.

The four attributes calculated from the Web pages of a session are:

1. The number of second-level syntactic topics of the session for each Web site, called SynTopic2PerSite;

2. The number of the session's page views for each Web site, called PVPerSite;

3. The number of second-level syntactic topics of the session for each first-level semantic topics of all the Web sites, called SynTopic2PerSemTopic1;

4. The number of the session's page views for each first-level semantic topic of all the Web sites, called PVPerSemTopic1.

The dimension of the description vector is equal to 2×*(the number of Web sites considered + the number of all first-level semantic topics for all the Web sites)*. Each attribute is normalized between [0..1] and has an importance weight that is assigned by the analyst depending on the context (the structure of the Web sites).

3.4.1.1 Session Summarization Example

Consider that we join log files from two Web sites: *www.inria.fr* and *www-sop.inria.fr*. Then, we structure the new log file in sessions like:
(a) http://www.inria.fr/actualites/index.fr.html
(b) http://www.inria.fr/actualites/colloques/index.fr.html
(c) http://www-sop.inria.fr/everest/events/cassis05/.

Two of the Web pages requested in this session belong to the Web site *www.inria.fr* and one to *www-sop.inria.fr*, therefore the second attribute is PVPerSite = (2 1).

The first-level semantic topics are: *"actualites"* for the first two pages and *"projets"* for the last page (i.e. "everest" is a team-project at INRIA). Suppose that the two Web sites have only 6 first-level semantic topics in total (including *"actualites"* and *"projets"*). In this case we have the PVPerSemTopic1 = (0 2 0 0 1 0).

The Web page *(a)* does not have a second-level syntactic topic, in contrast with *(b)* and *(c)* that have *colloques* and *events* as second level syntactic topics. As a result, the last two vectors have the following values for SynTopic2PerSite = (1 1), and for SynTopic2PerSemTopic1 = (0 1 0 0 1 0).

The size of the vector describing (summarizing) this session is $n = 2 \times (2 + 6) = 16$ and the vector is (2 1 1 1 0 2 0 0 1 0 0 1 0 0 1 0). The prototypes network of the neural algorithm (described in the following section) has, in this case, an input layer of dimension 16.

3.4.2 Neural Algorithm for the C&D's Clustering Step

We studied several clustering methods that handle sequences[3] and we describe here the most efficient method that we used for clustering sequences, introduced in [BT02].

The clustering method used in this research was based on a method developed in 2000 by [BT02] and integrated in the object-oriented framework CBR*Tools[4] [Jac98] for supporting the reuse of past experiences. It was successfully applied in the context of browsing advisors in a Web thematic repertory, for enterprises such as France Telecom. The efficiency of such a method is based on the neural approach and its effectiveness relies on the use of the summarized descriptions for sessions; in this case, the descriptions are based on a generalization of Web access sequences (cf. Section 3.4.1).

This method relies on a hybrid model of connectionist memory similar to the one described in [Mal96]. It is composed of a connectionist part inspired from [AG91, Gia92], which is a neural network based on prototypes with an evolutionary structure and on a flat memory compound of groups of patterns.

Definition 5 *A* **network based on prototypes** *is characterized by:*

1) A **memory of prototypes** *composed of vectors representing a sequence or a prototype which is memorized in the weights of the various network connections.*

2) A **learning mechanism** *which allows the memorization of a new sequence or prototype, or the modification of the existing prototypes. Such a mechanism acts on the weight values of certain connections.*

3) An **utilization mechanism** *allowing the use of the network for the* **remembrance** *of similar sequences; for example, for a new sequence presented to the network the network gives as output the class of the rememorized prototypes.*

The *structure* of a prototype-based network is evolutionary in the sense that the number of prototypes at the hidden level is not apriori fixed and can be increased during the learning step. A prototype is characterized by its reference vector, an influence region and a set of representative sequences.

3.4.2.1 Network Architecture

A prototype-based network has three levels, as described in the Figure 3.7 below [Gia92]:

1. The **input level** relaying on N units, N corresponding to the input space dimension (related to the sequence description).
2. The **hidden level** composed by the prototypes described in the same description space as the sequences; for example, the prototype j will be represented by the weight vector $W_j^{ec} = (w_{1j}^{ec}...w_{Nj}^{ec})$.

[3] In this paper, we consider user sessions, visits, episodes or sequential patterns extracted from such structures as **sequences** of Web pages or (generalized) Web pages.

[4] http://www-sop.inria.fr/axis/software.html

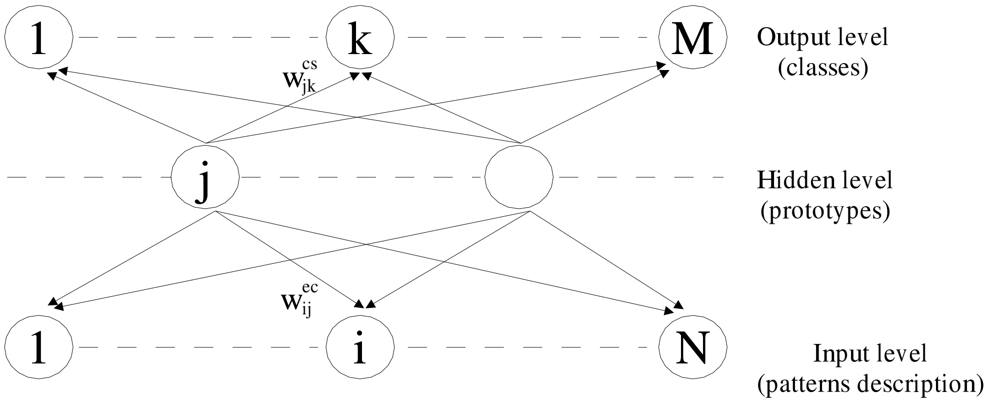


Figure 3.7: General Prototype-Based Network Architecture

3. The **output level** composed of M units corresponding to M classes; $w_{jk}^{cs} = 1$ of the prototype j belongs to the class k; $w_{jk}^{cs} = 0$ otherwise.

A threshold s_i is associated to each prototype, and it will be modified during the learning step. The threshold determines the *influence region* in the input space. If a sequence introduced in the network falls into the influence region of a prototype, then this prototype will be activated. Such a region is determined by the set of input vectors satisfying a distance measure lower than the threshold. If there is no activated prototype, a new one is created.

3.4.2.2 Activation Mechanism

Let m be a sequence that we need to classify. Let U be the set of prototypes having m in their influence region. The activation of the prototypes is given by the following equation :

$A_c = 1 \; if \; D(m, W_c^{ec}) = \min_{j \in U} D(m, W_j^{ec}) : A_c = 0 \; otherwise$

where c is the winner prototype, $W_c^{ec} = (w_{1c}^{ec} ... w_{Nc}^{ec})$ and D is an Euclidian distance (with normalization).

The weights w_{jk}^{cs} between the hidden level and the output level are given by the following formula:

$w_{jk}^{cs} = 1 \; if$ the prototype j belongs to the class k; $w_{jk}^{cs} = 0 \; otherwise.$

The activation of the class k (the case where O_k=1) is: $O_k = \sum_{j \in U} A_j \times w_{jk}^{cs}$

3.4.2.3 Learning and Memorization Modes

During the learning process, a sequence having the description vector $(m_1, m_2 ... m_N)$ is presented to the network. This sequence is added to the network by using the following algorithm:

1. If the sequence falls inside *one*, *several* or *no* influence region(s), the prototype closest to m is activated. This prototype is called the winner prototype C. In this case, maximum one class will be activated.

2. If the sequence m does not fall inside an influence region, a new prototype representing such a sequence is added on the hidden level.

3. The weights of the winner prototype are modified as follows:

   $W_c^{ec} = W_c^{ec} + \alpha(t) \times (m - W_c^{ec})$, where $\alpha(t)$ is a time-decreasing suite given by the following relation:

   $\alpha(0) = 1$; $\alpha(1) = c_1$; $\alpha(t+1) = \frac{\alpha(t)}{1+c_2\times\alpha(t)}$, where c_1 and c_2 are two constants.

   Such an operation allows us to obtain representative prototypes for each class.

A specific treatment based on a fuzzy region proposed in [Gia92] was also implemented for the boundary patterns, thus reducing the number of prototypes created at the hidden level of the network.

3.4.2.4 Neural Algorithm Illustration

Example 3 *Let us consider a group of students requesting the pages of a course about data mining:* `lambda/teaching/STID/`*. This course is located on the Web site hierarchy corresponding to the "lambda" project. Several pages can be requested from that part of the site:* `annee02-03.html`*,* `ex1.html`*,* `ex2.html`*,* `ex3.html`*,* `example-accesslog.html` *and* `example-errorlog.html`*. Each of these pages will be named a, b, c, d, e and f (i.e. a=*`lambda/teaching/STID/TD1.html`*, b=*`lambda/teaching/STID/TD2.html`*, etc.). Our goal is to detect that these six pages will be grouped in the same cluster by our algorithm. In fact, once this cluster is detected, a sequential pattern mining process on the sub-log file containing the sessions corresponding to that cluster will allow us to find the patterns with high support on this sub-log and very low representativeness on the entire log.*

Let us assume that we fusion logs from two Web sites, `www.inria.fr` and `www-sop.inria.fr`, and we structure such logs into sessions. Suppose that we obtain the following four sessions: <(a)(b)>, <(b)(c)>, <(d)(e)> and <(d)(f)>. The Web pages "a, b ... f" composing these sessions belong to the same Web site, `www-sop.inria.fr`, and all the Web pages have "projets" as their first-level semantic topic ("lambda" is a research project) and "teaching" as their second-level category. Hence, they have the same summary on the following four attributes:

```
- SynTopic2PerSite        0, 2
- PVPerSite               0, 2
- SynTopic2PerSemTopic1   0,0,0,0,2,0
- PVPerSemTopic1          0,0,0,0,2,0
```

The prototype network built on such an example has an input space dimension equal to 16 (related to the description of a sequence). Since the four sessions have the same description, we obtain only one class:

$C = \{< (a)(b) >, < (b)(c) >, < (d)(e) >, < (d)(f) >\}.$

This cluster will contain sessions corresponding to a small part of the Web site. Therefore, because of the high similarity of these sessions and because they will involve a limited number of items (the Web pages corresponding to that specfic part), the sequential pattern mining algorithm will extract patterns with high support inside the cluster. The same kind of patterns, when compared against the initial log, will have a very low support and would be impossible to obtain directly by an sequential pattern mining process on the initial file.

3.4.3 C&D Software Implementation

The C&D method is implemented as part of our toolbox for Web Usage Mining, the AxisLogMiner. The program reuses a previous implementation (in C) of the PSP algorithm developed by F. Masseglia [MPC99]. We implemented the neural algorithm in Java and we created a Perl script to split the structured log file into sub-logs based on the results obtained through the neural clustering. More details about this program are given in Section 4.2.

The program was tested on a real data set collected from INRIA's Web sites and the results obtained are described later, in Section 3.7.

3.5 Iterative Approach Illustration: Divide & Discover Method

We instantiated our Iterative Approach in the Divide & Discover (D&D) method. The D&D method aims at obtaining small and homogenous clusters of sessions. This will allow an sequential pattern mining process to discover interesting sequential patterns on these clusters. Because of the small size of the clusters, the patterns will have a very low support with respect to the entire structured log file, but a high support value inside the cluster, thus making them representative for a group of users.

The Figure 3.8 illustrates this method. First, by using the **PSP algorithm**, sequential patterns are extracted from the initial structured log file. Then, these patterns are clustered into n clusters (C_1 to C_n) by using the **neural algorithm** presented in Section 3.4.2. Next, the log is divided into n sub-logs (SL_1 to SL_n) upon these clusters and finally, a special sub-log (SL_{n+1}) is created for the sessions that could not be matched with any of the clustered patterns.

The quality of the results produced by our approach rely highly on the content and the size of this last sub-log. In fact, the first n sub-logs contain the most representative categories of users (e.g. users that browsed the main page, then the research teams

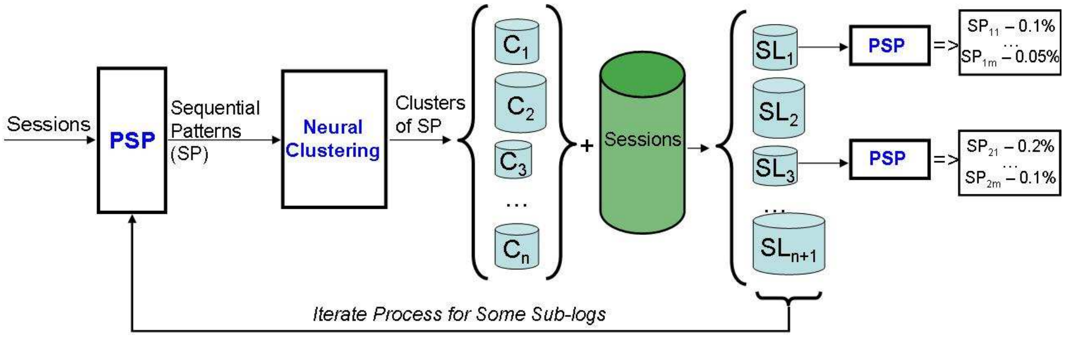


Figure 3.8: The D&D Method

page). They are representative, but the most interesting patterns will come from the study of the unclustered sessions of the sub-log SL_{n+1}. Considering this sub-log as a new original log, and repeating the process (as described in Figure 3.8) will allow us to discover behaviors with a low representativeness. The Figure 3.9 illustrates this concept.

Let us consider the unclustered sessions from SL_{n+1}. We copy these sessions in the sub-log $SL_{n+1_{k+1}}$. This process is reiterated until we find behaviors having a very low global support and a high representativeness inside the cluster.

Analyzing sessions from INRIA's log files allowed us to find a number of interesting behaviors as illustrated in Figure 3.9. We describe here the results obtained in one of our experiments while more details about these experiments are provided in Section 3.7. In the first iteration, the most represented behaviors found in INRIA's logs, are those consisting in pages from the following parts of the Web site:

- **/travailler/opportunites/** – representing the job opportunities part;
- **/recherche/equipes/** – describing the research teams;
- **/inria/organigramme/** – containing INRIA's flow chart;
- **/valorisation/logiciels/** – presenting the software developed at and available from INRIA.

Then, the iteration on the sub-log SL_{n+1} allows us to discover the behaviors related to some particular research teams such as *epidaure, oasis, rodeo* or *koala*. In the next iteration, the selection of the unclustered sessions at this step (the sub-log $SL_{n+1_{k+1}}$) shows behaviors related to the research teams (such as *mimosa* or *robotvis*), but also to the staff personal pages (e.g. *turletti*). Finally, this iterative process allows us to discover behaviors with a very low support compared to the original log. These behaviors are related to the research teams (*axis, planete*) or even to hacking activities (*/scripts/win32/...*).

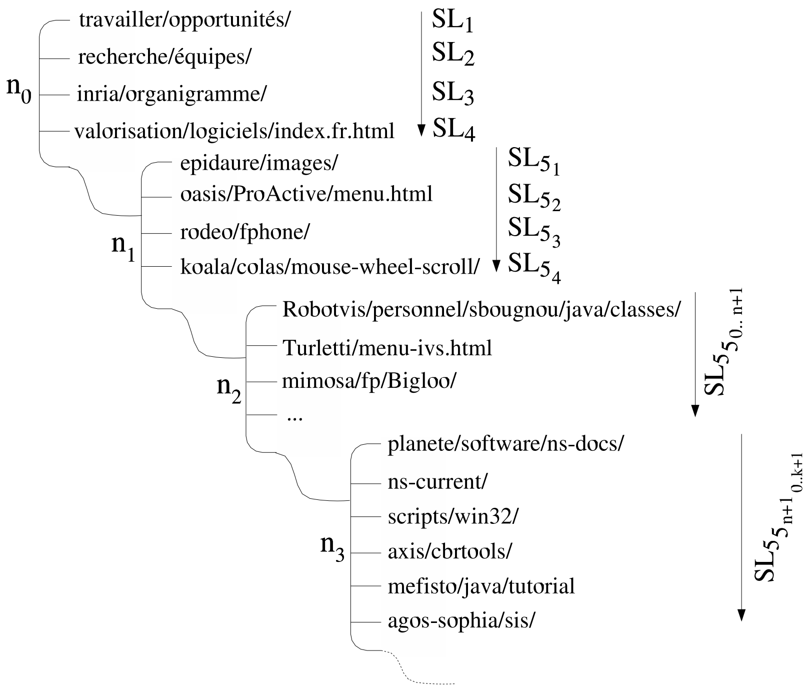


Figure 3.9: Recursive Division of a Log File with the D&D Method

In order to provide reliable results, our D&D method depends on the quality and accuracy of the division method proposed for a log. This division relies on the clustering performed on the sequential patterns discovered from the structured log file.

The neural clustering algorithm that we use in the C&D method to cluster sessions (see Section 3.4.2) is directly applicable on sequential patterns as they are composed of Web pages too. We use the summarization step described in Section 3.4.1, also in the case of the sequential patterns. However, the number and the size of attributes in the generalization method can be increased if more detailed semantical descriptions are available.

The efficiency of this algorithm is proven by the Example 3 presented in Section 3.4.1, where extracted sequential patterns can be considered instead of the sessions. These similar sequential patterns are grouped into a cluster which will generate a new cluster of sessions containing these short patterns. From the final cluster, longer and thus, more interesting patterns will be obtained.

3.5.1 D&D Software Implementation

The D&D method is implemented as a software tool integrated in our toolbox for WUM, the AxisLogMiner.

The program reuses a previous implementation (in C) of the PSP algorithm developed by F. Masseglia [MPC99], algorithm also used in the C&D method.

We use the same neural algorithm as for C&D (developed in Java) and we created a new Perl script to split the structured log file into sub-logs based on the result of the neural clustering. A detailed description of this program is provided in Section 4.3.

The program was tested on a real data set (from INRIA's Web sites) and the results are described later, in Section 3.7.

3.6 Hierarchical Approach Illustration: Hierarchical Discovery Method

The Hierarchical Discovery method (Figure 3.10) follows the Hierarchical Approach presented in Section 3.3.4. The sequential pattern mining algorithm that we use in this method is the Apriori-GST, that we describe in detail in the next section. This algorithm discovers Contiguous Sequential Patterns (noted CSPs), which are in fact sequential patterns with consecutive items. The CSPs have the property that their Web pages are located consecutively and without any gap in the supporting sessions. This new constraint in the SPM step permits the discovery of more precise patterns. They will generate smaller sub-logs giving to the data mining analyst the possibility to discover high representative patterns in these sub-logs.

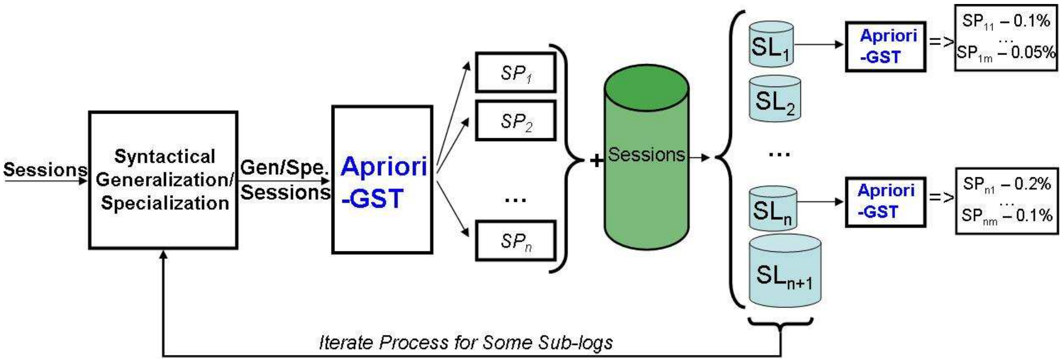


Figure 3.10: The H&D Method

3.6.1 Mining Contiguous Sequential Patterns with Apriori-GST

In this section, we present the Apriori-GST algorithm that we used to extract contiguous sequential patterns in the H&D method. The algorithm is an original combination between a Generalized Suffix Tree index and a modified version of the Apriori algorithm that uses this index for calculating the support of a candidate sequence.

We start by describing the principles of a suffix tree index, then we present the generalized version that we used in our method and, finally, we present the Apriori-GST algorithm.

3.6.1.1 Suffix Tree Index

Basically, a Suffix Tree (ST) is a data structure used for text indexing. Such an index is mainly used to search for a sub-string in linear time. This search is made possible by an initial treatment of the text which is also realized in linear time. We give a more complete definition as follows.

Definition 6 *Let S be a string, $S = x_1x_2...x_n$. A suffix of S is $S[i,n] = x_ix_{i+1}...x_n$. The Suffix Tree T for S is a tree with n leaves such that:*

- *There is a one-to-one relationship between any leaf of T and a suffix of S, $S[i,n]$. This leaf is labelled with i.*
- *The edges are labelled with non-empty words.*
- *The degree of its internal nodes is higher than 1.*
- *For a particular node, all children's labels begin with a different letter.*
- *The concatenation of the edges' labels of the path from the root node to a leaf i forms the suffix $S[i,n]$ of S.*

Hypothesis: No suffix is a prefix of another suffix [BGR01].

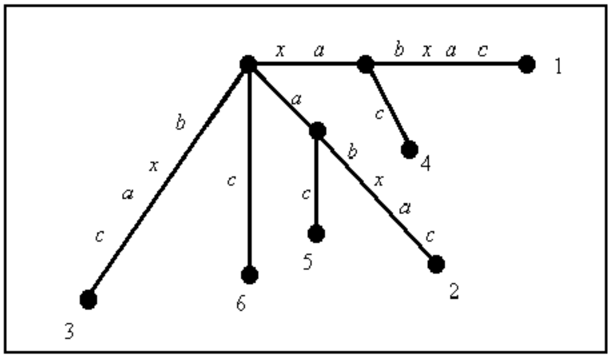


Figure 3.11: A Suffix Tree for the String `xabxac` [Gus97]

We have taken from [Gus97] the example of the suffix tree for the string `xabxac` (cf. Figure 3.11). The path from the root to the leaf number 1 is `S[1, n] = xabxac`. The path from the root to the leaf number 5 is `S[5, n] = ac`.

Without the final hypothesis, the definition of the suffix tree does not guarantee that we will have a corresponding suffix tree for each string. For example, if we consider the string ``xabxa'', the sequence ``xa'' is, at the same time, a suffix and a prefix for this string. Therefore, we cannot build a corresponding suffix tree for this string, according to the definition we have just announced. In order to solve this problem, we must add a new character, generically noted $, at the end of the string. This guarantees that no suffix is also a prefix.

There are several algorithms for building a suffix tree in O(n) time [Gus97]. We use the Ukkonen algorithm [Ukk95] for its good time and memory usage performances. For this algorithm, the construction of the ST is incremental and counts n steps, one for each suffix $S[i, n]$ of S. We do not give here the description of the algorithm for building a ST, but the interested reader can refer to [Gus97] for a complete and rigorous description.

3.6.1.2 Generalized Suffix Tree Index

According to [Gus97], a *Generalized Suffix Tree index* is a suffix tree index for at least two strings. The algorithm that builds the corresponding GST index for a set of strings is presented in this section.

Notations: The following notations are used in the algorithm:

- `N` is the total number of strings to be indexed.
- $\mathtt{S}_i$`, (i=1,..,N)` represents the i-th string.
- `T(i)` is the GST for the set $\{S_j \,|\, 0 < j < i + 1\}$.
- `R(T(i))` is the root node of $T(i)$.
- `v(i, e)` is an internal node of the tree (other than $R(T(i))$), where i is the number of the string that was indexed when the node was created. The edge of the node v, noted e, is represented by the pair (e_1, e_2), where e_1 is the index where the label of the edge e begins, in the string S_i, and e_2 the index where e ends.
- `l(i, e, `$\mathtt{P}_l$`)` is a leaf of the tree, where i and e have the same meaning as for an internal node. $P_l = \{(i_1,\, j_1),...(i_k,\, j_k)\}$ is the set of the suffixes represented by this leaf. We say that a suffix $S_i[j, n]$ is represented by the leaf l if the factor[5] of the node l perfectly matches $S_i[j, n]$. In this case, the pair (i, j) belongs to P_l.

Using these notations, we give the following two steps of the algorithm used for building a GST index:

`Step 1`: We build $T(1)$, the ST for S_1, using the Ukkonen algorithm.

[5] The factor of a node v is the string formed by the concatenation of all the labels of the edges in the path from $R(T(i))$ to v.

`Step 2`: For any string S_i, $0 < i < N+1$, we traverse in the current GST following the path S_i as far as possible. Let us suppose that we arrive at position j in S_i (this means that the first $j-1$ characters from S_i are contained in the current GST). We obtain $T(i)$ by applying the Ukkonen algorithm from step j until all the suffixes $S_i[\mathrm{p}..n_i]$, $p > j-1$ are added to $T(i-1)$.

It is possible for several strings to have a common suffix. In this case, the corresponding leaf $l(i, e, P_l)$ will contain this information in the set P_l. The set P_l of $l(i, e, P_l)$ contains all the pairs (i, j_i), where j_i represents the beginning of the common suffix in S_i.

In Figure 3.12, we show the GST obtained by adding the string "`babxba`" to the ST for "`xabxa`". Using this tree, we can determine that the string `xba`, for instance, is contained in the second string indexed (i.e. `babxba`) starting with the position 4. If we are interested in the string `bx`, then we follow the `b` and `x` edges in the tree and list all the pairs (sequence, position) found in the leaves. In this way, we determine that the string `bx` is included in both strings at position 3.

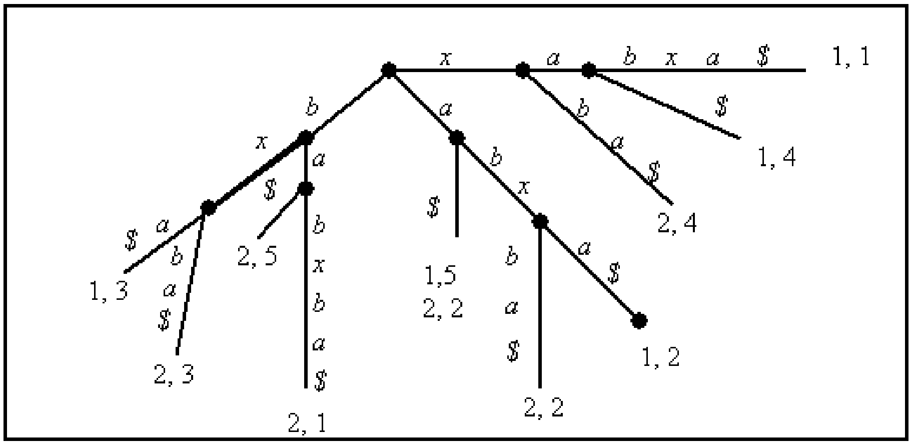


Figure 3.12: A GST for the Strings S_1="`xabxa`" and S_2="`babxba`"

3.6.1.3 Sequential Patterns Discovery with Apriori-GST

We propose Apriori-GST, an Apriori-like algorithm, that uses a GST index for discovering contiguous sequential patterns. We remind that the support for a sequence S is defined as the ratio between the number of sequences containing S and the total number of sequences. A *frequent sequence* (sequential pattern) is a sequence whose support is superior to a minimum support, called *minsupp*. In our method, *minsupp* is the only parameter that has to be input by the analyst.

In the WUM context, a sequence is defined as follows: let $I = \{i_1, i_2, ..., i_m\}$ be the set of all IDs or items (Web pages). A sequence S is a set of items ordered by their time stamp and noted $S = (s_1\, s_2 \ldots s_n)$. A k-sequence is a sequence of k items. The sequence S is a sub-sequence of another sequence $S' = (s'_1\, s'_2 \dots s'_m)$, (with $n < m$) if there are two positive integers j, k (with $n = k - j + 1$), s.t. $s_1{=}s'_j$, $s_2{=}s'_{j+1}$,...,$s_n{=}s'_k$.

In order to determine the frequent sequences, the Apriori-GST algorithm tests, at each step k, all the k-sequences from the set C_k (candidate k-sequences). This set is filtered using the minimum support, *minsupp*, and we obtain the set of frequent k-sequences, L_k (sequences having the support > *minsupp*). A join on L_k gives the C_{k+1} set used in the next step. The initial C_1 set is formed by all the items of I and the algorithm stops at step k, when C_k is empty. As we can see, the test for determining the support of a k-sequence is performed very often. In the Figure 3.13, we give the recursive function `supp` that we use to calculate the support for a k-sequence. Let us note that $v.child(s_1)$ is the child of v, which is introduced by the edge that begins with s_1[6]. The function `suffix(S, nextNode)` removes at the beginning of S the items from the edge between v and *nextNode*. $v.ds$ gives the total number of distinct sequences found in all the sets P_l, where l is a leaf of the sub-tree that has v as the root. $v.ds$ is calculated after the construction of the GST for each node and its value is updated when new sequences are added.

```
Function supp(S, v)
// Input: $S = (s_1\, s_2 \dots s_k)$ a k-sequence, $v = (i, e, child, P_v)$ a node of the GST
// Output: the sequence's support (a real value between [0,1]).
    if ($v$ is null) then return 0;
    else
        if ($S$ is empty) then return $v.ds/N$;
        else
            $nextNode = v.child(s_1)$;
            return supp(suffix(S, nextNode), nextNode);
end supp;
```

Figure 3.13: The Function **supp** Used for Calculating the Support of a Sequence

3.6.1.4 Originality of Our Apriori-GST Algorithm

The Apriori-GST algorithm (presented in Figure 3.14) has two major differences from the classical Apriori approach. First, for calculating the support of a candidate k-sequence, noted c, we use the GST index by calling the `supp` function provided previously with the root node of the GST ($R(T)$) as the first argument.

Second, in the main Apriori iteration, when calculating the set of frequent sequences of length k (L_k), we compute directly the support for all candidates c of C_k without checking against all the sequences S from the database. No access to the disk is required for this operation as the GST index is kept in the main memory.

[6] We remind that, for an internal node in a suffix tree, all children's edges begin with a different item.

```
Procedure Apriori-GST (S, I, minsupp, T(N))
// Input: $S$ the set of $N$ sequences, $I$ the set of items, $minsupp$ the minimum support,
        $T(N)$ the GST
// Output: the set $L$ of frequent sequences
$k = 1$; $C_1 = I$;
while ($C_k \neq \phi$) do
    for each ($c \in C_k$) do
        if ($\mathbf{supp}(c, R(T)) > minsupp$)
        then $L_k = L_k\ U\ \{c\}$;
    $k = k + 1$;
    GenerateCandidate ($C_k$, $L_{k-1}$);
end while;
return $L = U_{j=1}^{k}\ L_j$
end Apriori-GST;
```

Figure 3.14: Apriori-GST Algorithm

The Apriori-GST algorithm we introduced is an Apriori-like algorithm because it uses the Apriori general principle and the candidate-generation mechanism. Unlike the other Apriori approaches and especially GSP [SA96] or PSP [MPC99], which extract sequential patterns, Apriori-GST aims to **discover only contiguous sequential patterns**. However, this limitation allows us to employ the GST index presented previously, with three main advantages:

1. The database of sequences (or the structured log file) is read only once and is loaded into the main memory. The algorithm uses linear time and space to build and, respectively, store the sequences [Gus97].

2. The time necessary to calculate the support of a candidate k-sequence is also a linear ($O(k)$), thanks to the GST index.

3. The GST index is easily updateable, a very useful property, as the content and usage of a site frequently changes (new pages are added, old pages are removed). New sessions are added daily to the log files and thus old sessions, containing removed pages, must be deleted. As we can see from its construction, the GST algorithm is **incremental**, therefore, new sessions are easily added to the current GST index. Removing an old session is a similar operation but implies the removal of the corresponding leaves for all the suffixes of a sequence.

3.6.1.5 Validation of Apriori-GST and H&D Implementation

The Apriori-GST algorithm was successfully used to extract sequential patterns from a structured log file [TT01]. However, in that study, it was directly applied on the sessions. In the Hierarchical Discovery method, the algorithm is used on the generalized sessions, thus allowing it to index (in the GST structure) and, then, to process a significantly greater number of sequences (because of fewer distinct items).

Moreover, we used the Apriori-GST algorithm to extract contiguous sequential patterns from time-series gene expression data obtained from differentially expressed genes during the development of mouse cerebellum [TLT04]. We provide the results of this study in Appendix B.

We are currently working at the software implementation of the H&D method, therefore, no results with this method are currently available.

3.7 Experiments

The main objective which drove our experiments was to show that, by using the approaches we presented before, **efficient** and **effective** methods can be instantiated. We used as examples, the two methods implemented, the C&D and D&D.

In this section, we show first that C&D is more efficient than the classical sequential pattern mining algorithms such as WAP-mine [PHMAZ00] and PSP [MPC99] when mining sequential patterns with very low support. The efficiency is measured both in terms of execution times and in terms of the lowest support value for which we are to discover sequential patterns.

Second, by presenting some of the behaviors discovered by using C&D and D&D, we show that these methods are capable of finding interesting sequential patterns with low support, patterrns that cannot be found using classical methods (as detailed in Section 3.7.3.1).

3.7.1 Datasets

The log files that we used in our experiments were collected from INRIA's main Web server (`www.inria.fr`) and from INRIA Sophia Antipolis Web server (`www-sop.inria.fr`) during one month, February 2003. The total size of the two files was 2.4 GB. We had slightly more than two thirds of the requests for the main Web site `www.inria.fr` and the other third for `www-sop.inria.fr`. In Table 3.3, we summarize the main characteristics of the initial dataset.

Characteristic	`www.inria.fr`	`www-sop.inria.fr`	**Total**
Size of the log file	1,607 MB	837 MB	2,444 MB
Number of requests	9,139,391	4,193,695	13,333,086
% of requests	68.54%	31.46%	100%

Table 3.3: Description of the Initial Log Files

On the raw log files, we applied the preprocessing methodology described in Chapter 2. After the preprocessing step, the size of the structured log containing user sessions and visits was reduced to only 332 MB. This represents a total of 382,625 visits, from which only 165,359 contained at least 2 pages (cf. Table 3.4). We selected this set of visits with at least two pages, to be used as input for the sequential pattern mining applications.

Characteristic	**Value**
Size of the structured log file	332 MB
Number of sessions	244,293
Number of visits	382,625
Number of long visits (length $\geq$ 2 pages)	165,395

Table 3.4: Characteristics of the Structured Log File

The tests were conducted on a Linux powered computer with a bi-processor Pentium IV, running at 2.8MHz and with 2 GB of main memory. The results of the tests are described in the following section.

3.7.2 Efficiency Analysis for the C&D Method

For the efficiency analysis, we compared the C&D method, which implements the Sequential Approach, with WAP-mine [PHMAZ00] and PSP [MPC99]. We used an implementation of WAP-mine realized by T. Zhu[7]. As in the WAP-mine algorithm, an itemset may be composed of only one item, we modified the PSP algorithm to treat the input data similarly. Also, the output of the WAP-mine algorithm consists in sequential patterns which are not pruned (i.e. subsequences of a sequential pattern discovered are also output). In order to compare methods that have similar inputs and outputs, we also needed to modify the PSP algorithm and remove the pruning phase from it. The version of the PSP algorithm, which included the two changes was called PSP^{-} and it was also used within the C&D method.

We compared the three methods by extracting the sequential patterns with very low support (i.e. the lowest support values for which the algorithms terminate) from the dataset described before. We varied the support value from 0.1% to 0.01% and we measured the response time for the three algorithms as presented in Figure 3.15. From this, we observe that C&D acts as a complement for the “classical” methods of sequential pattern mining. For very low support values (0.02% to 0.06%), WAP-mine and PSP^{-} are not able to extract any patterns. In contrast, C&D finds 1,652 patterns for a support value of 0.048% and 77,045 patterns for 0.025%. Below 0.02%, the execution time for C&D also grows exponentially. However, some interesting patterns are found only by C&D, like for example: SP2 involving pages about the Free Phone software, SP3 revealing hacking activities on INRIA’s Web site or SP8 with pages about the meetings of the internal committee (cf. Section 3.7.3.1).

[7] http://www.cs.ualberta.ca/∼tszhu/software.html

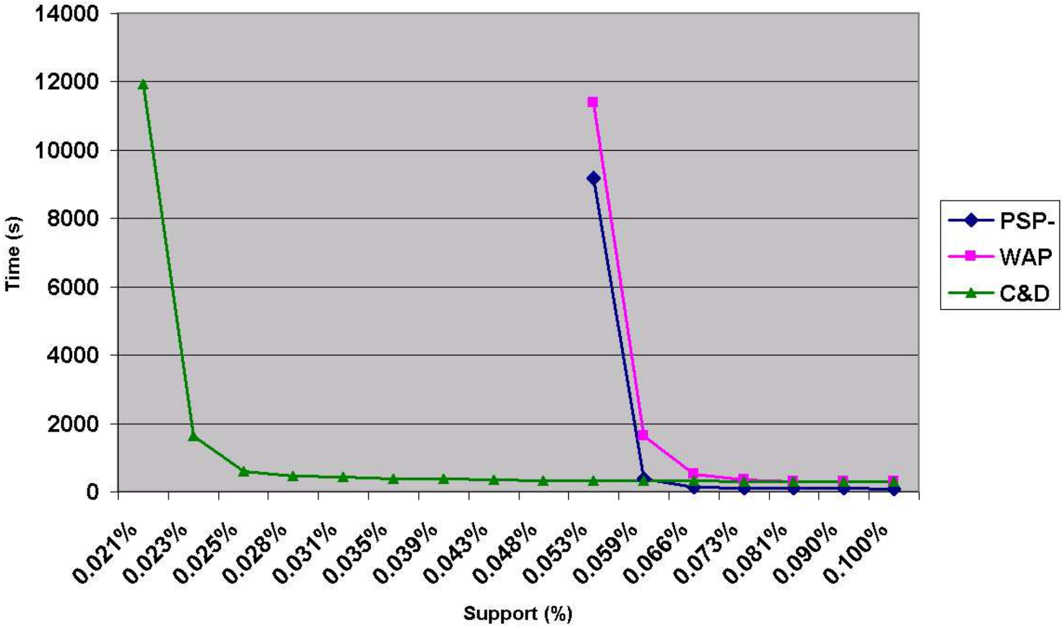


Figure 3.15: Execution Times for WAP, PSP^- and C&D

For the clustering step of the Cluster & Discover method, we needed to define the weights of the vectors describing a session. We used the following values: W(PVPerSite) =0.25, W(SynTopic2PerSite)=0.25, W(PVPerSemTopic1)=0.75 and W(SynTopic2PerSem Topic1)=0.75 and we obtained 155 clusters with this configuration.

3.7.3 Comparative Analysis: C&D and D&D

In the second experiment conducted, we extracted frequent behaviors using both methods, the C&D and D&D. We were interested in finding the sequential patterns with very low support that will correspond to a minor (small) group of users. For patterns with such low support values, it would not be possible to use a classical sequential pattern mining algorithm as it would either generate thousands of results or would not terminate (as, for instance, when using 0.021%, the support for SP3, as the minimum support value).

Eight patterns (SP1 – SP8) are listed in Table 3.5. For each pattern, the value of the *local support*, i.e. the support of the pattern in the sub-log where it was discovered, is given in the **LS** column. The LS figures are higher for the D&D method because more iterations will generate smaller clusters. The *number of iterations* in the D&D method is provided in the column **NI**. The final *global support*, i.e. the support of the pattern with respect to the entire structured log file is given on column **GS**. This value is the same for both methods, the C&D and D&D, as the two were applied on the same dataset.

	D&D		**C&D**	
Id	LS	NI	LS	GS
SP1	30.7%	3	8.5%	0.074%
SP2	2.06%	3	0.47%	0.055%
SP3	85.7%	9	2.56%	0.021%
SP4	28.0%	2	9.69%	0.201%
SP5	15.0%	3	1.51%	0.159%
SP6	5.53%	4	3.46%	0.022%
SP7	48.1%	4	1.54%	0.056%
SP8	26.6%	3	0.59%	0.027%

Table 3.5: Comparison of Results for D&D and C&D

As mentioned in Section 3.5, both methods use the same neural clustering algorithm and same sequential pattern mining algorithm. The D&D extraction process was analyst-driven, and therefore we had to choose manually, at each iteration, the support for the sequential pattern mining algorithm in that iteration. For the D&D method, depending on the sequential pattern, we needed several iterations (cf. Table 3.5) for finding some of the behaviors listed below.

3.7.3.1 Examples of Patterns Discovered

The following are some of the sequential patterns discovered using our both methods:

SP1: with the prefix **http://www-sop.inria.fr/mascotte/personnel/Sebastien.Choplin/cours/iut-infocom/excel/**: <(exercices.html) (exo1.xls) (exo2.xls) (exo3.xls) (exo4.xls) (exo5.xls)>. This pattern reflects the behavior of the users interested in the teaching activities of S. Choplin. This group of users was probably constituted by students who took Mr. Choplin's course.

SP2: with the prefix **http://www-sop.inria.fr/rodeo/**: <(fphone/) (fphone/features.html) (fphone/obtain.html) (fphone/start.html)>. This pattern describes the navigation of the users wanting to find more information about the Free Phone, an audio tool for Internet telephony developed by the Rodeo research team at INRIA Sophia Antipolis. We see that the users first look at the page describing the Free Phone, then they examine its features and how they could obtain it, to finally look at the way of installing the application.

SP3: < (http://www.inria.fr/MSADC/root.exe?/c+dir) (http://www.inria.fr/scripts/%2e%2e%255c%2e%2e/winnt/system32/cmd.exe?/c+dir) (http://www.inria.fr/_mem_bin/%2e%2e%255c%2e%2e/%2e%2e%255c%2e%2e/%2e%2e%255c%2e%2e/winnt/system32/cmd.exe?/c+dir) (http://www.inria.fr/scripts/%2e%2e%c1%1c%2e%2e/winnt/system32/cmd.exe?/c+dir) (http://www.inria.fr/scripts/%2e%2e%c1%9c%2e%2e/winnt/system32/cmd.exe?/c+dir) (http://www.inria.fr/scripts/%2e%2e%25%35%63%2e%2e/winnt/system32/cmd.exe?/c+dir)>. This behavior is typical of a search for a security hole in the system. The INRIA's system administrators confirmed this when we asked them about these results. This is another pattern with very low support that would have been impossible to detect using a direct sequential pattern mining process.

SP4: with the prefix **http://www.inria.fr/**: <(travailler/opportunites/chercheurs .fr.html) (travailler/opportunites/chercheurs/concourscr2.fr.html) (recherche/equipes/ index.fr.html) (recherche/equipes/listes/index.fr.html)>. This behavior is related to the career opportunities offered by INRIA. The users read the job opportunities page describing the "researcher" position, then the page describing the competitive selection and, finally, the pages describing the research teams.

SP5: with the prefix **http://www.inria.fr/**: <(/) (travailler/opportunites/it.fr.html) (travailler/opportunites/ita/missions.fr.html) (travailler/opportunites/ita/concoursit.fr .html) (travailler/opportunites/ita/faqita.fr.html)>. This is another pattern related to the careers section of the Web site. This time, the users are interested in the technician job opening, the recruitment process and the job requirements. We found several patterns related to the careers section of the Web site because every year in February, the new job openings are published on INRIA's Web site.

SP6: with the prefix **http://www-sop.inria.fr/mefisto/java/tutorial1/**: <(tutorial1.html) (node3.html) (node4.html)>. This is not a very interesting pattern, but confirms that users browse (in order) chapters from a Java tutorial.

SP7: with the prefix **http://www.inria.fr/**: <(/) (recherche/index.fr.html) (valorisation/logiciels/index.fr.html) (valorisation/logiciels/calcul.fr.html)>. This pattern, although quite interesting, has a very low global support. It concerns the users that looked first at the research activities available at INRIA, then at the software developed by our research teams, to finally focus on the page containing the tools for "scientific computing" (`valorisation/logiciels/calcul.fr.html`).

SP8: with the prefix **http://www-sop.inria.fr/interne/**: <(vie/) (vie/comites/cp/) (vie/comites/cp/2003/cp130203/)>. This is a pattern related to the intranet Web pages of INRIA Sophia Antipolis, which explains its very low global support (0.026%). The pattern contains pages about INRIA's internal events, pages about the internal projects' committee and their meeting organized in February 2003.

The eight behaviors presented are specific for users that came and visited INRIA's Web site driven by different objectives. All these patterns represent the behaviors of small groups of users. The number of users following them is very low compared to the total number of visits, in some cases representing less that 50 visits (SP3, SP8). Therefore, discovering them by a classical sequential pattern mining would have been impossible. On the contrary, with the C&D (automatically) and D&D (interactively), we were capable of finding the patterns presented above and we could emphasize the effectiveness of our method.

3.8 Related Works

Several data mining techniques have been applied to Web Usage Mining, so far:

- Association rules [CMS99, ZXH98, BGG$^+$01],
- Sequential patterns [MPC99, SFW99, BGG$^+$01, MTP01a, XD01, ESRR04],
- Clustering applied on: sessions [Kum04, VBYA04, NK02, MDLN02, FSS00], users [PPK$^+$00, CHM$^+$00, DG01] or pages [MDLN02, SBP04],
- Decision trees [ZXH98, BGG$^+$01] and
- Time series [ZXH98].

From those, the sequential pattern mining algorithms are particulary well suited for WUM, given the sequential aspect of the users' sessions. However, as shown in the previous section, classical sequential pattern mining algorithms such as WAP-mine [PHMAZ00] and PSP [MPC99] cannot find sequential patterns with very low support when applied on a large structured log file.

As detailed in the previous sections of this chapter, our methods instantiating the three approaches we proposed for Intersites WUM combine both the existing sequential pattern mining and the clustering techniques. In the Figure 3.16 enclosed below, we have represented the three methods together with the relevant WUM techniques:

- The H&D method (light blue) combining the sequential pattern mining algorithm "Apriori GST" with the URL generalization step and
- The C&D (green) and D&D methods (red) grouping together the sequential pattern mining algorithm PSP and the neuronal clustering techniques.

Based on this particularity of our methods, which combine two data mining techniques for improving the extraction of sequential patterns with low support, we have studied and compared them with other works that are also employing the two techniques, together or separately, for mining sequential patterns.

In Section 3.8.1, we compare the Apriori-GST algorithm used in the Hierarchical Discovery method with similar algorithms for sequential pattern mining that were used on Web usage data. We are interested in outlining the main differences between our algorithm and the existing methods. Some of these algorithms could be also used in new methods that instantiate the Hierarchical Approach or for the SPM step in the other two approaches.

As the clustering step is essential for both the Sequential and Iterative Approaches, we were interested in various clustering methods that were used on sessions. The most representative works are presented in Section 3.8.2.

According to our knowledge, only the ApproxMAP technique [Kum04] uses a similar strategy for discovering approximate sequential patterns. We describe this method in Section 3.8.3.

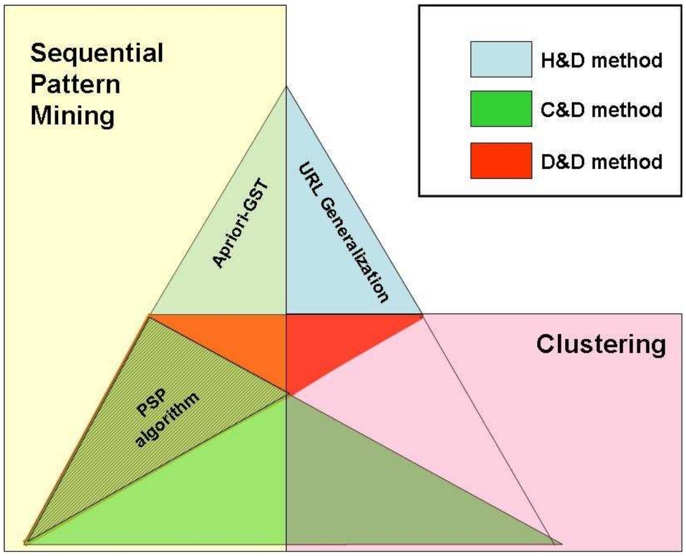


Figure 3.16: Diagram of Our Methods

3.8.1 Related Works in Sequential Pattern Mining

There are numerous works dealing with the sequential pattern mining techniques applied to WUM [Spi99, MPC99, PHMAZ00, XD01, ESRR04]. In this section, we describe the main contributions in this field and compare them with our Apriori-GST algorithm.

Some techniques, like [XD01], use an internal data structure similar to the one used in our Apriori-GST algorithm included in the H&D method. In Table 3.6, we compare these techniques with the Apriori-GST. We identified and listed the following properties for the output of these techniques:

- GSP (Generalized Sequential Pattern) - the algorithm discovers sequential patterns that are formed by pages which are not necessarily contiguous in the sequences supporting the sequential pattern;

- MFS (Maximum Forward Sequence) - the algorithm is capable of finding the maximum sequential pattern by pruning the frequent sub-patterns included in this MFS;

- CSP (Contiguous Sequential Pattern) - the sequential patterns extracted are contiguous in the sequences supporting them;

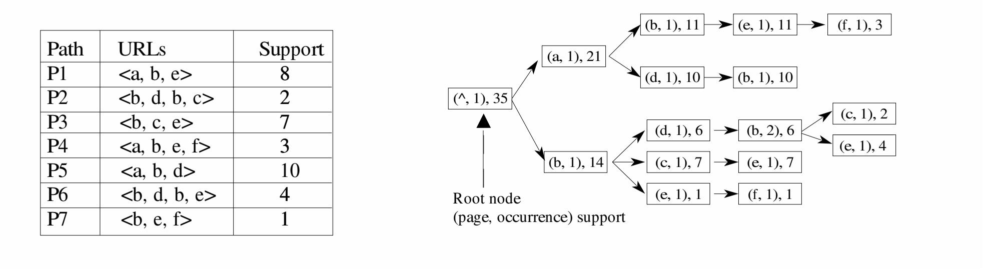
- DI (Duplicate Items) - duplicate (consecutive) items are accepted in a sequential pattern.

Name or Ref.	**SPM Technique**	**Data Structure**	**Properties of the Result**
WUM	Constraint-based	Aggregated Tree	GSP
WebTool	PSP (Apriori-like)	Hash Table	GSP, MFS
WAP-mine	PrefixSpan-like	WAP-tree	GSP
OAT	DFS	Suffix Tree	MFS, CSP, DI
FS-Miner	DFS	FS-Tree	MFS, CSP, DI
Apriori-GST	**Apriori-like**	**GST**	**MFS, CSP, DI**

Table 3.6: Comparison of Different Sequential Pattern Mining Techniques for WUM

3.8.1.1 Web Utilization Miner (WUM) [Spi99]

The Web Utilization Miner (WUM) tool aims to discover sequential patterns which are considered as "interesting" from a statistical point of view. WUM proposes to extract sequential patterns having a minimum support and matching a user-defined pattern. For this, the sessions are transformed into an aggregated tree. Each node in this tree is linked to a page from the session path. The authors then propose a prefix tree corresponding to the sessions as detailed in Figure 3.17.



Path	URLs	Support
P1	<a, b, e>	8
P2	<b, d, b, c>	2
P3	<b, c, e>	7
P4	<a, b, e, f>	3
P5	<a, b, d>	10
P6	<b, d, b, e>	4
P7	<b, e, f>	1

Figure 3.17: Example of an Aggregated Tree for Seven Sessions [Spi99]

The patterns extraction step is based on requests composed of pages and wildcards, written in the MINT language (an SQL-like language). The data mining analyst will request for instance, the number and the patterns of the form "$a * c$", where a and c are two Web pages. For this request, only the patterns beginning with a and ending with c will be returned. Thus, this method belongs to the constraint-based methods and it does not allow the discovery of "unknown" new patterns as our approaches do.

3.8.1.2 WebTool [MPC99]

In [MPC99], the authors propose a new system for WUM, the WebTool. This system takes into account all the steps of a WUM process, from the data selection to the results display, *via* the data transformation and patterns extraction. The WebTool is based on a prefix tree (the PSP, proposed by the authors) to extract sequential patterns. The goal is to obtain all the frequent patterns relying on the "generating pruning" method (Apriori principle). PSP, like other methods based on this principle, becomes less efficient when the minimum support or the sequential pattern representativeness is very low. The PSP algorithm is also used in our methods, the C&D and D&D, as detailed in Sections 3.4 and 3.5.

The PSP Algorithm

The WebTool uses the PSP algorithm introduced in [MPC99]. The algorithm is based on the same general algorithm as GSP [SA96], but it uses an improved tree-like data structure for storing candidate sequences. The *prefix-tree* data structure used in PSP contains all the candidates in the following way: any branch going from the root to a leaf stands for a candidate sequence, and considering a single branch, each node at depth i captures the i^{th} member of the sequence. Furthermore, along with each item, the support of the sequence from the root to that leaf node is also stored. For instance, the support of the candidate sequence $<$ (10), (20) $>$ is 2, in Figure 3.18. The PSP algorithm considers two cases for consecutive items: they can either belong to the same transaction (dashed line in Figure 3.18) or to different transactions. Unike the PSP, our algorithm, the Apriori-GST, considers that a transaction can have only one item.

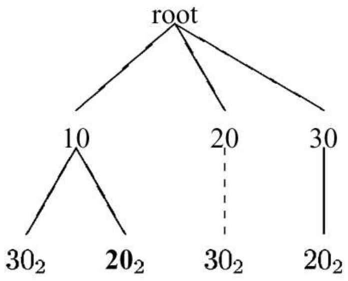


Figure 3.18: PSP Tree Data Structure [MPC99]

3.8.1.3 WAP-mine [PHMAZ00]

The WAP-mine, described in [PHMAZ00], is a method that allows the extraction of frequent patterns from the user sessions, refereed as "Web access patterns" (WAP) in this paper. The algorithm builds an aggregated structure called WAP-tree. By using this structure, the authors overpass the need for candidate generation as in other Apriori-like algorithms, thus making the approach more efficient.

We compared our C&D method to the WAP-mine and PSP (cf. Section 3.7) and we showed that for sequential pattern extraction for low support only the C&D method is capable of extracting sequential patterns.

3.8.1.4 Online Adaptative Traversal [XD01]

The authors of [XD01] were interested in discovering contiguous sequence patterns in a Web log file; their approach is similar with our Apriori-GST algorithm. During the preprocessing step, they included backward traversals in the sessions (contrary to the MF method [CPY96], presented in Section 2.5.3).

The sequential patterns are called **Maximal Frequent Sequences (MFS)** in their case and they represent frequently used contiguous sequences of page references. In this context, the definition of a subsequence is: *X is a subsequence of the sequence A if all the terms of X are found in A consecutively. E.g. if $A = \langle a, b, c, d\rangle$ then $\langle a, b\rangle$ and $\langle b, c\rangle$ are subsequences of A and $\langle a, c\rangle$ is not.*

When comparing this method with our work, we can see that their algorithm, the OAT (Online Adaptive Traversal), used for mining MFS is based on a suffix tree, while our algorithm, the Apriori-GST, uses a generalized suffix tree (i.e. a suffix tree for several sequences). Moreover, OAT searches for sequential patterns by doing an extensive depth-first search in the suffix tree, while the Apriori-GST uses the candidate generation principle and checks the support only for candidate patterns.

OAT was design to fit the suffix tree in the main memory, i.e. it is an adaptive algorithm that uses two pruning techniques and compresses the suffix tree. It was shown that the suffix tree takes linear main memory with respect to the total length of the big sequence [Ukk95] and the memory space needed by the OAT's suffix tree is bounded by $36N + 4n$, where N is the total length of the *big sequence* (i.e. sequence obtained by concatenating all the sequences) and n is the number of the sequences.

For instance, the authors extract sequential patterns with a minimum support as low as $s_{min} = 0.01\%$. However, according to their experiments, the longest MFS actually consists of 79 reloads of the same page and therefore their results are rather uninteresting.

Compared with the H&D method, the OAT is a direct application of an sequential pattern mining algorithm on the structured log file. On the contrary, in the H&D method, we apply the Apriori-GST algorithm in several iterations, where the semantic/syntactic topics of the pages are considered at different hierarchical levels. Both algorithms use the same data structure (the suffix tree), but the Apriori-GST requires less steps for finding the support of the candidate sequences than the OAT.

3.8.1.5 FS-Miner [ESRR04]

The FS-Miner algorithm [ESRR04] is based on the FS-Tree which is a compressed tree used to represent sequences. Unlike the generalized suffix tree index, the sequences are compressed and only partly included in the FS-Tree to reduce the memory space occupied by the data structure. The database is first scanned to identify frequent links (frequent sequences of size 2) and then the FS-Tree is build incrementally.

Mining the FS-Tree involves a depth-first traversal, which is less time consuming than in the OAT algorithm, as the FS-Tree is smaller. However, the FS-Miner does not clearly outperform the Apriori-based algorithms (such as Apriori-GST) as stated by the authors in [ESRR04].

3.8.2 Clustering Techniques

Numerous clustering methods have been used for data mining purposes. However, only a few have been applied to the Web Usage Mining. This is due to the fact that adapting these methods to the Web data is difficult and sometimes even impossible.

One of the main difficulties consists in finding an accurate distance measure between sessions. The quantity of data (number of sessions or number of pages) can also be a problem because most clustering methods are memory based. In order to overcome these problems, various techniques were used in [MDLN02, NK02, SBP04, VBYA04, FSS00]. A high-level summary of the main clustering techniques described in this section is given in Table 3.7, while more details are provided below.

Name or Ref.	**Clustering Technique**	**Objects Clustered**	**Input Data Structure (DS)**	**Similarity Measure**	**Output DS**
PACT [MDLN02]	Multivariate k-means	Sessions	Binary Vector	Cosine Coefficient	Aggregated Usage Profiles
H-UNC [NK02]	Hierarchical Clustering (HC)	Sessions	Binary Vectors	Syntactic Similarity	Ses. Profile Vectors
SUGGEST [SBP04]	Incremental Connected Components Algorithm	Web Pages	Weighted Site Graph	Link Weight	Clusters of Web Pages
[VBYA04]	SOFM	Sessions	Vector	Defined	Clusters
[FSS00]	BIRCH (HC)	Generalized Sessions	Vector of “Time on Page”	Euclidean Distance	Clusters of Sessions

Table 3.7: Comparison of Different Clustering Techniques for WUM

3.8.2.1 PACT [MDLN02]

In [MDLN02], the authors used PACT (Profile Aggregations based on Clustering Transactions), a multivariate k-means clustering method, for grouping sessions in similar clusters. The PACT technique builds aggregated profiles from Web sessions without taking into account the order of the requests. The sessions (“transactions” in this paper) are represented by binary vectors balanced by page views. The weight for a page can be either the value of a time function or a function that considers the type of the page. PACT groups the users' sessions based on a similarity function applied on the binary vectors of the sessions. The results of this method are expressed as aggregated usage profiles. An aggregated usage profile is composed of couples (page view, weight).

The PACT technique was employed for providing recommendations as part of a personalization engine [MDLN02].

3.8.2.2 H-UNC [NK02]

In [NK02], the authors describe a hierarchical clustering technique that uses notions of genetic biological niches, called Hierarchical Unsupervised Niche Clustering (H-UNC). The sessions are encoded as an N-dimensional binary attribute vector, where the *ith* attribute, $s_j^{(i)}$, from a session is 1 if the user accessed the j^{th} URL and 0, otherwise. The similarity measure employed is based on the amount of the overlap between the URL's paths taken two by two. This is a "syntactic similarity" as it relies on the site structure.

This clustering algorithm works as follows: an initial population of randomly selected sessions is coded into binary vectors. This population competes using a density based fitness measure that has the highest value at the center of good (i.e. dense) clusters. Different niches in the fitness landscape correspond to distinct clusters in the data set. Once the clustering phase is over, the sessions from a cluster are summarized in terms of a session profile vector. The components of this vector represent the URL's relevance weights and measure the relevance of an URL to that specific profile.

The H-UNC could be used as an alternative method in our clustering step, although it may be difficult to apply it on large Web logs for Web sites with huge number of URLs. Especially in the Intersites WUM case, the sessions are very sparse and calculating the sessions similarity for H-UNC would require a significantly long time.

3.8.2.3 SUGGEST 2.0 [SBP04]

In [SBP04], the authors propose a page-clustering method that uses the Incremental Connected Components algorithm and a graph model for the Web site map. The vertices are represented by the Web pages, while the edges are represented by the weighted links between the Web pages. The weight for a link between two pages i and j is calculated as: $W_{ij} = N_{ij}/max\{N_i, N_j\}$, where N_{ij} is the number of sessions containing both pages i and j, N_i is the number of sessions containing the page i and N_j is the number of sessions containing the page j. The *visit coherence* formula they use reduces the weight for the so-called "index pages" (i.e. pages containing mainly links for other pages). Using the Incremental Connected Components algorithm, clusters of Web pages are built based on this link weight. Therefore, this approach is complementary to ours as it identifies clusters of Web pages and not clusters of sessions.

3.8.2.4 Velasquez & al [VBYA04]

In [VBYA04], the authors used a Self-Organizing Feature Map (SOFM) algorithm to cluster the user sessions. Each session is modelled as an input vector v of the pages and the time spent on a page. They defined a similarity measure between the sessions that takes into account the page content, the time spent on a page and the sequences of pages for the two sessions.

The drawback of using SOFM is that vectors of the same size, H, are required as input. If a session has fewer elements than the H, the rest of the elements are filled with 0. For vectors greater than H, only the first H elements are considered. This signifies that some potential interesting sessions are truncated.

The authors tested their algorithms on a real dataset, but the number of different pages was relatively low (217), compared with the number of pages available from a large Web site.

3.8.2.5 Generalized Sessions Clustered with BIRCH [FSS00]

In [FSS00], the sessions are generalized by means of an induction based on the attributes. This induction reduces the data dimensions. For instance, **www-sop.inria/lambda/teaching/stid-projet2.html** is organized as a hierarchy such as the following: **www-sop.inria → lambda → teaching → stid-projet2.html**.

The authors then use BIRCH [ZRL96], an incremental hierarchical clustering algorithm, to cluster the data generalized. The method has been tested on a log file collected from the University of Missouri-Rolla Web server. The log contained 2.5 millions lines, but the tests were conducted only on 500,000 lines. The number of users was 26,107 and the number of pages was 21,203. But, upon the authors, BIRCH gets less efficient when data dimension is increased, hence the data generalization is limited to a few levels.

3.8.3 Combining Sequential Pattern Mining and Clustering

Like our two methods (C&D and D&D), the ApproxMAP [Kum04] combines clustering and sequential patterns for multiple alignment sequential pattern mining. The authors propose the extraction of multiple alignment sequential patterns. Their objectives and results are different from ours, as they extract one approximative sequential pattern for each cluster.

ApproxMAP uses KNN clustering with weighted edit distance to group similar sequences into clusters. The edit distance (or Levenstein distance) [Gus97] counts the number of "edit" operations (*insert, delete, replace*) to transform a sequence S_1 into another sequence S_2. The distance between two sequences is normalized by dividing it by the length of the longest sequence [Kum04].

For each cluster obtained, the method calculates a weighted sequence. In Table 3.8, there are 4 sequences (seq_1 to seq_4) with a maximum length of 4 itemsets. For these sequences the following weighted sequence is calculated incrementally: $<(A:3,\ E:1):3\ \ (B:1):1\ \ (B:4,\ C:3,\ X:1):4\ \ (D:4, E:2):4>$. This weighted sequence is composed of four weighted itemsets and represents the alignment of the four sequences. The weight of the first itemset is 3, representing the number of sequences that have an itemset in the first position. The items A and E have the weights 3, and respectively 1, corresponding to the number of sequences that have A/E in their first itemset.

The pattern consensus sequence (last row of Table 3.8) is computed from the weighted sequence for a threshold of three sequences (minimum support or "cut-off point"). This pattern summarizes the content of a cluster (set of sequences) into a single approximate sequence.

SeqID	**Itemset1**	**Itemset2**	**Itemset3**	**Itemset4**
seq_1	$< ()$	$()$	(BC)	$(DE) >$
seq_2	$< (A)$	$()$	(BCX)	$(D) >$
seq_3	$< (AE)$	(B)	(BC)	$(D) >$
seq_4	$< (A)$	$()$	(B)	$(DE) >$
$wseq_3$	$< (A:3, E:1):3$	$(B:1):1$	$(B:4, C:3, X:1):4$	$(D:4, E:2):4 >$
Pattern ($w \geq 3$)	$< (A)$		(BC)	$(D) >$

Table 3.8: Example of an Weighted Sequence [Kum04]

There are two main distinctions between AproxMAP and our C&D and D&D methods:

- We extract, using an sequential pattern mining algorithm, *ALL* the sequential patterns (with a support greater than a threshold) from a cluster, while the AproxMAP only calculates a general pattern for that specific cluster.

- Our objective is to find sequential patterns with very low support, representing the browsing activities for a small group of users with similar interests on the Web site, while their objective is to describe a cluster based on only one approximate sequential pattern. This is rarely possible for Web usage data as, usually, we have several behaviors (although similar) represented inside the same cluster.

Finally, we think that, due to its summarizing capabilities (only one sequence per cluster), the AproxMAP can be used as an alteranative method to get an overview of the analyzed dataset.

3.9 Conclusions and Perspectives

In this chapter, we presented a general divisive methodology for extracting sequential patterns. This methodology completes the classical sequential pattern mining methods by its ability in discovering patterns at low support values from log files. Based on this divisive methodology, we developed three approaches for WUM: the Sequential Approach, the Iterative Approach and the Hierarchical Approach. We instantiated these approaches into methods and corresponding software tools, called "Cluster & Discover" (C&D) – for the Sequential Approach, "Divide & Discover" (D&D) – for the Iterative Approach and "Hierarchical Discovery" (H&D) – for the Hierarchical Approach.

We studied the efficiency of our C&D method by comparing it to the WAP-mine and PSP on two log files of INRIA's Web sites: `www.inria.fr` and `www-sop.inria.fr`. We have found that some interesting patterns are discovered only by the C&D method, while the other two methods are unable to extract patterns with such a low support.

In the near future, we plan to extend these experiments to other datasets, coming from different types of Web sites (portals, e-commerce, intranets, etc.).

The main advantage of our approaches is that they can be easily instantiated by following the general steps of each approach. For instance, in the Sequential Approach, one clustering method and one sequential pattern mining technique are needed to obtain a new method based on this approach. We intend to build and compare several methods for each of these approaches, in order to find the most efficient combination.

The future works will also include a study on the "utility" of the patterns extracted to validate our methods' effectiveness. For this, we will need to clearly define a series of *utility criteria* (such as *patterns that contain pages from different semantic topics, or pages not being directly linked are more interesting than others*).

An open issue with the actual clustering method employed in the C&D and D&D methods is that it uses semantic topics[8]. For the moment, these have to be manually defined before the analysis. However, we consider that the current research in the Semantic Web field (see Section 5.2.7) will influence the development of more structured Web sites, followed by Web mining applications for these Web sites [BHS02], in the future.

Furthermore, we intend to ameliorate the neural clustering algorithm that we are currently using in the C&D and D&D methods, by adopting an improved version of the session summary. This new session summary will also take into account the "utility" of the patterns extracted.

[8] Actually, the sessions generalization step in the D&D method can be also done based on the syntactic topics.

Chapter 4

AxisLogMiner Toolbox for WUM

To support our global methodology for WUM, we designed and implemented the AxisLogMiner toolbox as a collection of tools for Web Usage Mining. This software toolbox contains several applications for the preprocessing step, the sequential pattern extraction and visualization of the Web usage data.

Currently, we integrated the following applications in the AxisLogMiner toolbox:

1. A preprocessing tool, the **AxisLogMiner Preprocessing**, which takes as input several log files corresponding to several Web servers. The output of the preprocessing consists in one structured log file containing sessions, visits and episodes. If the user decides to store the preprocessing data in our relational model, then a MySQL database is created and additional variables are computed and stored in this database (see Section 2.6.3). The results of the preprocessed log files can be further analyzed with our methods for sequential pattern mining (described in Chapter 3) or other statistical tools. For instance, the preprocessed Web log data described in our first experiment presented in Section 2.7.1 was analyzed using the SAS package.

   We used Perl scripts to implement the data fusion, data cleaning, and data structuration steps as described in Chapter 2. For the user interface and data summarization, we used Java and SQL.

2. A first automated sequential pattern extraction tool, the **Cluster & Discover**, corresponding to the C&D method described in Section 3.4. This application is programmed in Java and uses a previously developed algorithm, implemented in C together with conversion scripts coded in Perl.

   This application takes as input the structured log files output by the AxisLogMiner Preprocessing. The output of the application consists in frequent sequential patterns found in the clusters as explained in Section 3.4.

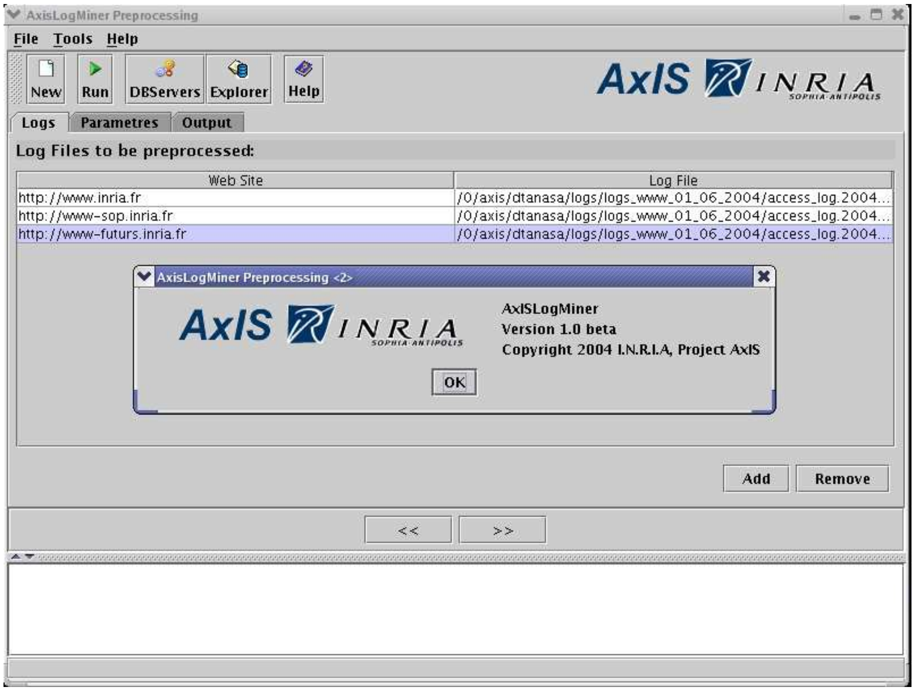
3. A second interactive SPM tool, the **Divide & Discover**, corresponding to the D&D method described in Section 3.5. As for the C&D application, we also used

Java and Perl for implementing this application. The input and output formats for this application are similar with the ones used in Cluster & Discover.

We will present each of these three applications in this chapter as follows.

4.1 AxisLogMiner Preprocessing

The AxisLogMiner Preprocessing (Figure 4.1) is a software application, developed in Java, that implements our preprocessing methodology. We used Java to implement our application as this gives several benefits both in terms of added functionality and in terms of easiness of implementation. The application uses modules written in Perl for the operations carried on the log file. The Perl language has very good performances when working with text files, therefore we implemented as Perl modules the following operations: log files join, log cleaning, robot requests filtering and session/visit/episode identification[1]. To store the preprocessing log file, in our relational model we used JDBC with Java.



Log Files to be preprocessed:

Web Site	Log File
http://www.inria.fr	/0/axis/dtanasa/logs/logs_www_01_06_2004/access_log.2004...
http://www-sop.inria.fr	/0/axis/dtanasa/logs/logs_www_01_06_2004/access_log.2004...
http://www-futurs.inria.fr	/0/axis/dtanasa/logs/logs_www_01_06_2004/access_log.2004...

AxisLogMiner Preprocessing <2>

AxISLogMiner
Version 1.0 beta
Copyright 2004 I.N.R.I.A, Project AxIS

Figure 4.1: AxisLogMiner Preprocessing

The aim of AxisLogMiner Preprocessing (ALMP) is to prepare the raw Web server's log files for the data mining step of the Web Usage Mining process. In other words,

[1] The session identification module is a modified version of the WebLog script available at http://awsd.com/scripts/weblog/

this includes the log files cleaning and the requests grouping into sessions and visits. The cleaning process is customized by the user which can specify all the preprocessing parameters using the ALMP interface.

In this section, we present the steps that the analyst must follow in order to clean the Web server's log files.

4.1.1 Graphical User Interface

The Graphical User Interface (GUI) of the ALMP tool is shown in Figure 4.2. The use of the ALMP to perform the preprocessing of the HTTP log files (using WUM techniques) is straightforward.

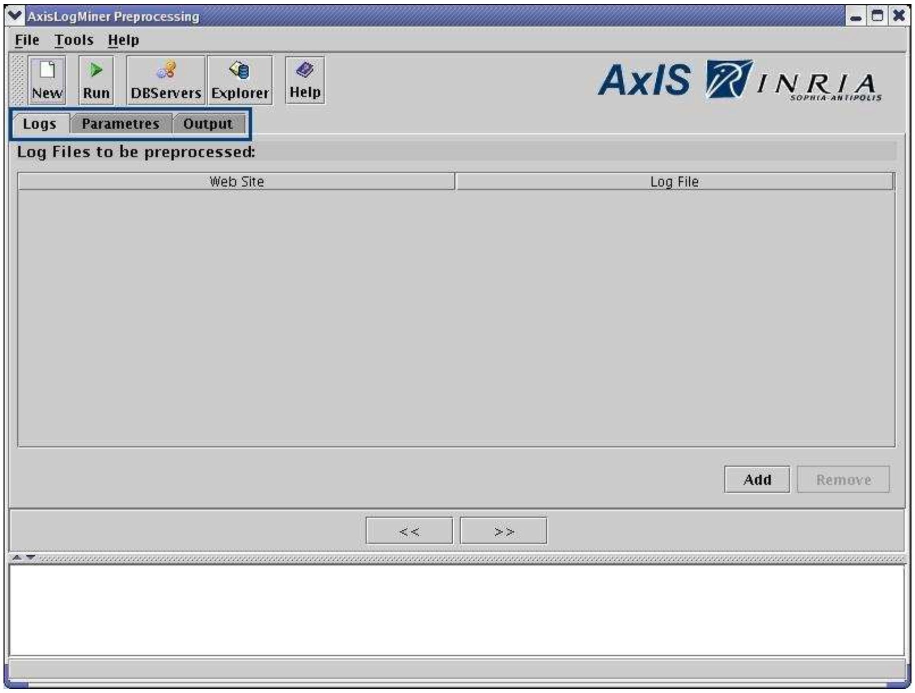


Figure 4.2: GUI of AxisLogMiner Preprocessing

The **Menubar** of the application contains three menus:

1. **File** which contains the following menu items:

   - **New Log Preprocessing** – prepares the ALMP for a new log preprocessing process; this means that the data previously entered into the main tabs (i.e. Logs, Parameters and Output) of the application will be lost and the default values will be shown.

- **Run Preprocessor** – validates the data entered in the main tabs and executes the preprocessing of the log files and the storage in the database if the user chooses to store the preprocessed data in a database.
- **Exit** – exits the application.

2. **Tools** which includes the following menu items:

   - **Database Servers** – shows the database connections manager tool (see Section 4.1.3.1) that allows the user to manage connections to different database servers. The DB servers can be used then to store the preprocessed data in a relational database.
   - **DB Explorer** – shows the database explorer tool (see Section 4.1.3.2) which allows the user to browse a relational database specified in the database connections manager tool and to execute SQL queries in order to analyze the data stored.

3. **Help** which is composed of two menu items:

   - **Topics** – shows the application help file.
   - **About** – shows the name and the current version of the application.

The **toolbar** also contains buttons which trigger the menu items actions.

4.1.1.1 Application Main Tabs

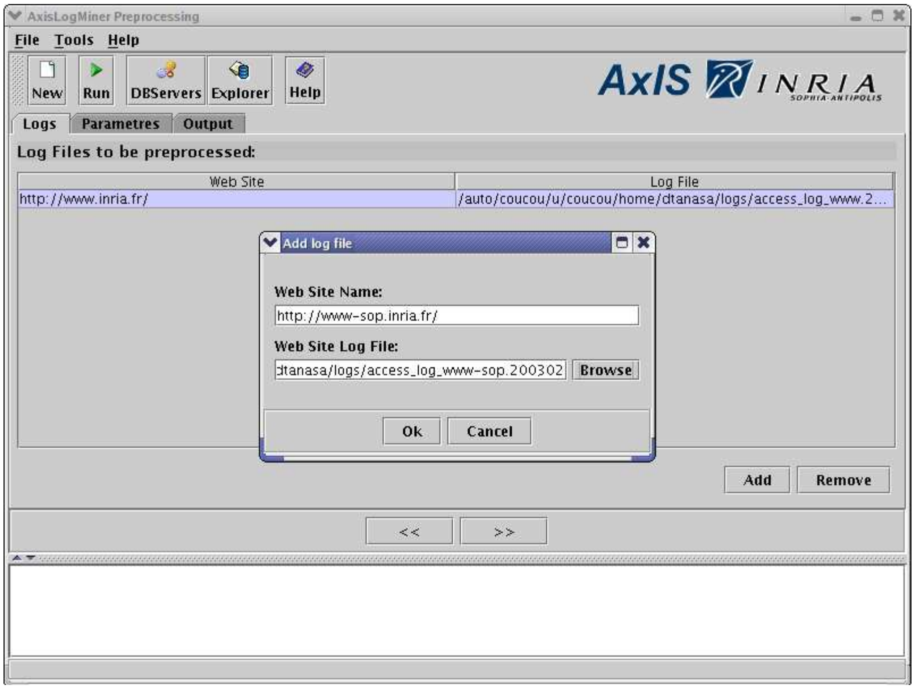
The application GUI contains three tabs as highlighted in the Figure 4.2. The three tabs are:

- **Logs** – allows the input of the log files to be processed by the ALMP.
- **Parameters** – allows the user to specify the preprocessing process parameters: the extensions of the files to be ignored (like image, multimedia and script files), the previously discovered hosts of Web robots which will also be ignored, the anonymize option (user selects whether to anonymize or not the IP addresses from the log files).
- **Output** – allows the user to specify where the output of the preprocessing process will be stored (i.e. the preprocessed log file, the file containing the hosts hosts that will be identified as being Web robots, the information related to anonymization, the database where the output is stored).

The navigation between these tabs can be done either by using the mouse or by using the two buttons situated on the top of the log window.

4.1.1.2 Application Log Window

The log window located at the bottom of the frame displays the on-going messages about the status of the preprocessing process and error messages (if errors occurs).



Log Files to be preprocessed:

Web Site	Log File
http://www.inria.fr/	/auto/coucou/u/coucou/home/dtanasa/logs/access_log_www.2...

Web Site Name: http://www-sop.inria.fr/

Web Site Log File: dtanasa/logs/access_log_www-sop.200302

Figure 4.3: Raw Log Files to Be Preprocessed

4.1.2 Using AxisLogMiner Preprocessing

The ALMP has an intuitive and user-friendly interface. From the main window of the preprocessing application, the user needs to perform several steps in the following order:

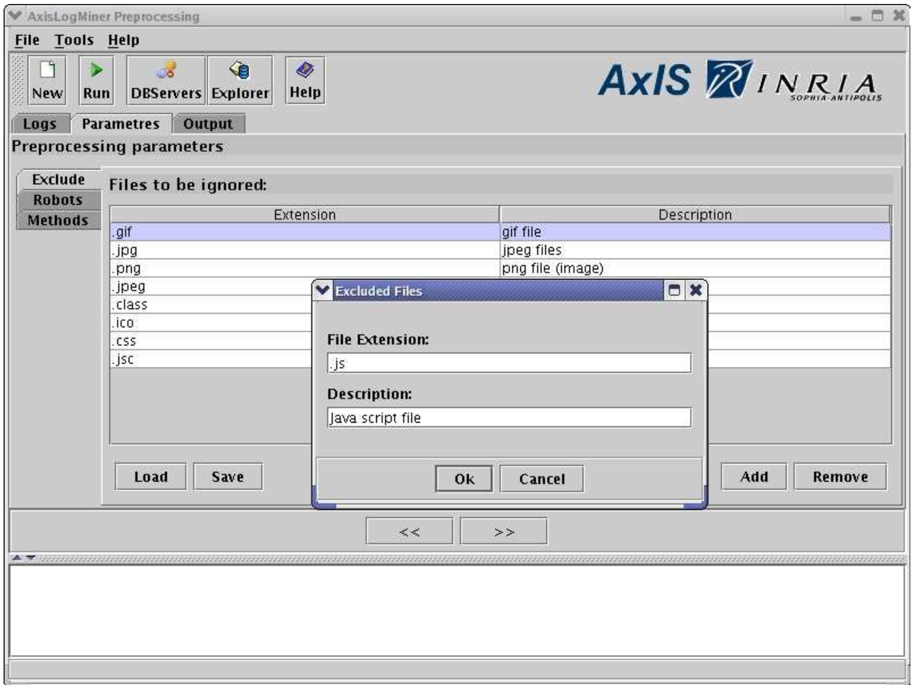
1. **Choose the files to be processed using ALMP**. The files must be available on the computer or network drive and the name of the corresponding Web server for each file must be known.

2. **Specify the preprocessing parameters**. Several preprocessing parameters can be set depending whether the analyst is interested in keeping or not the requests for specific types of resources.

3. **Specify the output format**. The path for the output files will be specified here. Also, if the analyst intends to save the preprocessed data in a database, the connection details to the DB must be specified.

4. **Execute the preprocessing process**. This is the last step which will generate the output based on the options previously selected.

All these steps are illustrated in the following sections.



Preprocessing parameters

Files to be ignored:

Extension	Description
.gif	gif file
.jpg	jpeg files
.png	png file (image)
.jpeg	
.class	
.ico	
.css	
.jsc	

Excluded Files

File Extension: .js

Description: Java script file

Figure 4.4: Preprocessing Parameters: Exclude Files

4.1.2.1 Choose the Files to Be Processed Using ALMP

The first step consist in choosing the Web server's log files to be preprocessed. From the application's **Log** tab the analyst can use the *Add* and *Remove* buttons in order to add or remove the raw log files.

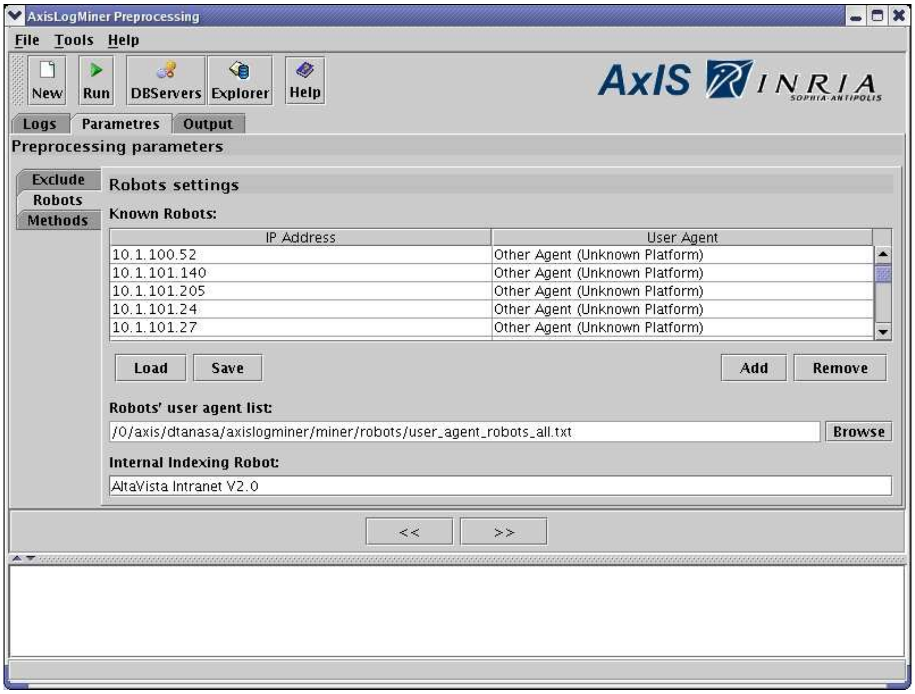
When a new log file is added, the user must specify the name of the Web server which generated the log file. For instance, in the Figure 4.3, we add a new log file for the Web server `http://www-sop.inria.fr`.

4.1.2.2 Specifying the Preprocessing Parameters

The next step consists in setting up the preprocessing parameters. In this step, the users can specify the non-analyzed Web requests. These requests will be excluded from the structured log file. Also, here, the users select the methods to be employed for robots detection (their requests will be removed from the log file), and whether or not to anonymize the IP's of the HTTP requests.

The set up of the parameters is done in the **Parameters** tab which includes the following 3 vertical tabs:

1. In the **Exclude** tab (Figure 4.4), the users can specify the Web requests that need to be filtered out depending on the type of the Web resource requested. The types



Robots settings

Known Robots:

IP Address	User Agent
10.1.100.52	Other Agent (Unknown Platform)
10.1.101.140	Other Agent (Unknown Platform)
10.1.101.205	Other Agent (Unknown Platform)
10.1.101.24	Other Agent (Unknown Platform)
10.1.101.27	Other Agent (Unknown Platform)

Robots' user agent list: /0/axis/dtanasa/axislogminer/miner/robots/user_agent_robots_all.txt

Internal Indexing Robot: AltaVista Intranet V2.0

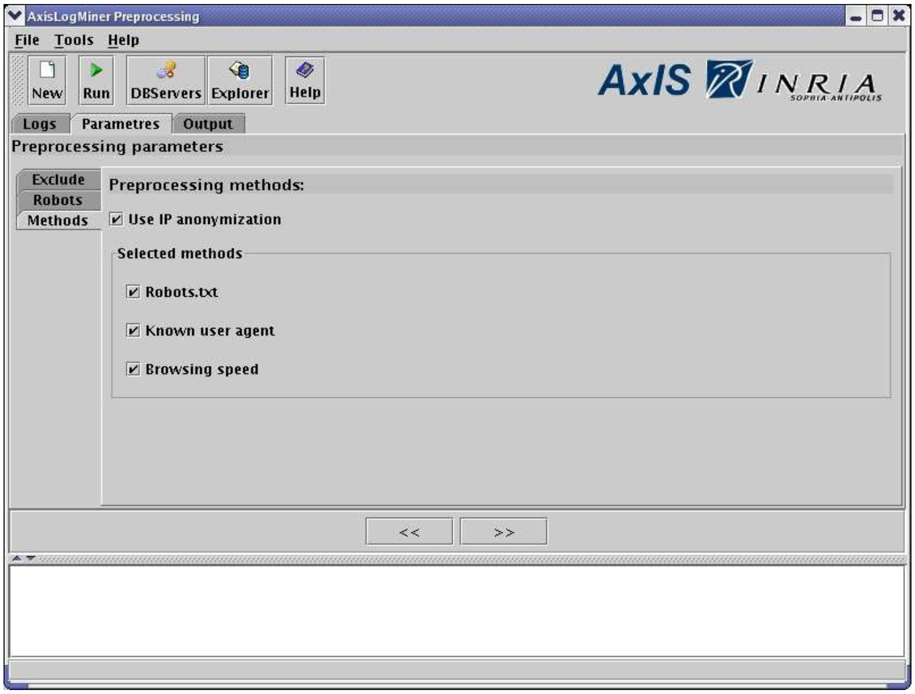
Figure 4.5: Preprocessing Parameters: Robots

of the files are specified through their extension. The *Add* and *Remove* buttons can be used to add or remove a file extension. The file extensions list can be saved to a file, for later use, using the *Save* button. Moreover, the file extensions list can be populated from previously saved files using the *Load* button.

2. In the **Robots** tab (Figure 4.5), the settings concerning the Web robots identification can be specified. All the requests made by the robots' hosts provided in this tab will be removed from the HTTP log file before the creation of the structured log file.

   The robots settings that can be specified here are:

   - The *Known Robots* – users can specify the IP and the user agent of the known robots' hosts. This list can be created during a previous preprocessing.
   - The *Robots' user agent list* – a file with the user agents known to be robots. There are several lists available, but the analysts can create their own list by adding the new user agents of the hosts identified as being Web Robots.
   - The *Internal indexing robot* – represents the Web site's internal crawler agent that is used to index the Web pages for the site search service. For instance, at INRIA, the "Altavista Intranet V2.0" robot is used to index the Web site's pages for the search function. Requests made by this WR should be eliminated from the log file.



Preprocessing parameters

Preprocessing methods:

☑ Use IP anonymization

Selected methods

☑ Robots.txt

☑ Known user agent

☑ Browsing speed

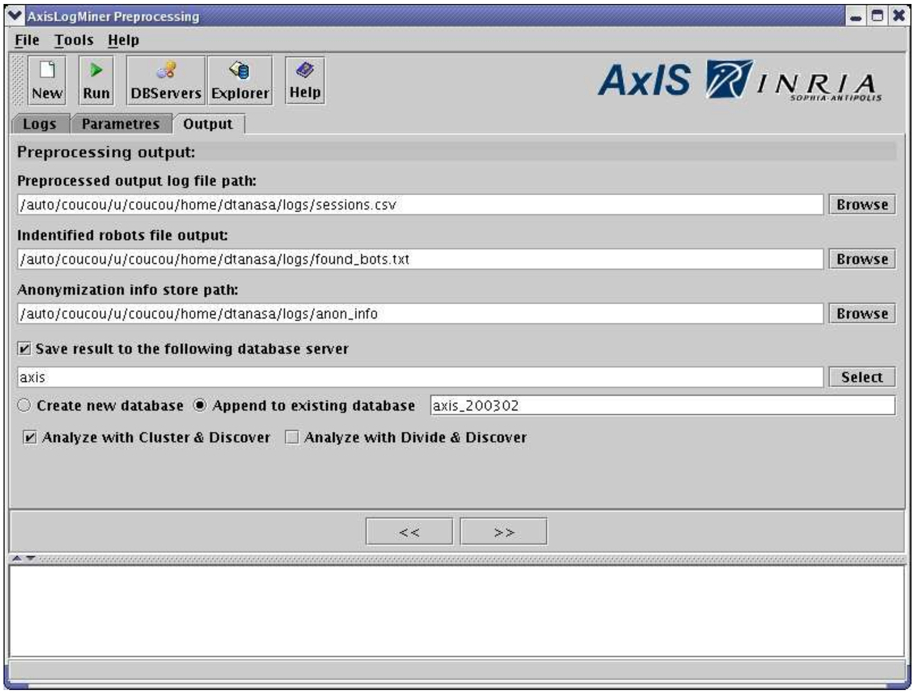
Figure 4.6: Preprocessing Parameters: Methods

3. In the **Methods** tab (Figure 4.6), the users can specify if the IP addresses will be anonymized or not. There are many privacy concerns regarding the log files, because when analyzing them, one can easily see the Web pages requested by another person. For this purpose, it is enough to know the host name or the IP address of this person. Because of these concerns, we chose to give the analyst the possibility to anonymize the host IP address or its name. The IP is replaced by an ID and the correspondence is kept in the analyst's database, in a separate table (i.e. the REAL_IP table). In this way, the analyst may share the preprocessed log file or database (without the REAL_IP table) without disclosing any confidential information.

   Moreover, in this tab, the users can choose the robot identification methods that will be used. There are three methods available for the moment: Robots.txt, Known User Agents and Browsing Speed (all described in Chapter 2). The users can choose to use any combination of the three methods for robot identification.

4.1.2.3 Specifying the Output Format

In the **Output** tab (Figure 4.7), the users specify the locations where the files generated by the preprocessing will be saved and what SPM software tools will be launched upon program completion. The result file containing the structured log will be stored



Preprocessing output:

Preprocessed output log file path:
/auto/coucou/u/coucou/home/dtanasa/logs/sessions.csv

Indentified robots file output:
/auto/coucou/u/coucou/home/dtanasa/logs/found_bots.txt

Anonymization info store path:
/auto/coucou/u/coucou/home/dtanasa/logs/anon_info

☑ Save result to the following database server
axis

○ Create new database ● Append to existing database axis_200302

☑ Analyze with Cluster & Discover □ Analyze with Divide & Discover

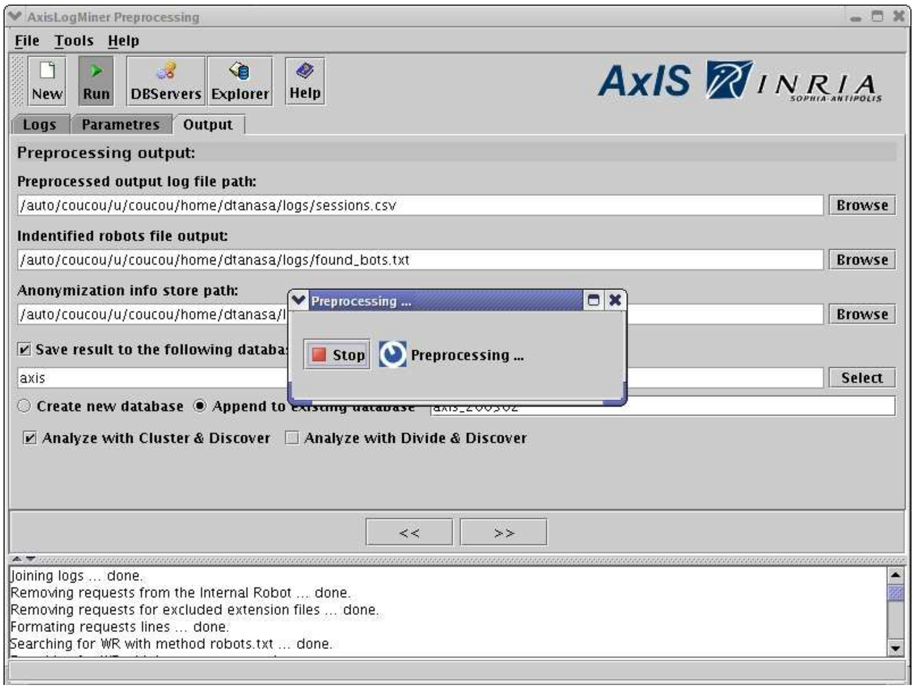
Figure 4.7: Output Settings

in the location entered in the *Preprocessed output log file path* box. In the location indicated by the *Identified robots file output* box, will be stored the IPs and user agents for the new robots discovered during the preprocessing.

If the users decide to anonymize the IPs, they have to input a directory name in the *Anonymization info store path* box, and in this directory will be stored the mappings between the original IPs and the anonymized IPs (they can also be stored in the database if this form of output is selected).

The users can also choose to save the structured log file in a relational database by checking the *Save result to the following database* checkbox. If they select this option, a database connection will have to be provided using the *Select* button which will pop up the Database server connections manager (see Section 4.1.3.1). A valid database name must be provided, in this case. The data can be stored either in a new database or appended to an exiting database (see Figure 4.7). The structure of the relational database is shown in Chapter 2, Figure 2.5.

In the **Output** tab the users may also specify the SPM application to start after the preprocessing. At the moment, they may choose between the “Cluster & Discover” and “Divide & Discover” tool as shown in the Figure 4.7. The chosen application will be launched after the preprocessing with the preprocessing output file selected as input file for the SPM step.



Preprocessing output:

Preprocessed output log file path: /auto/coucou/u/coucou/home/dtanasa/logs/sessions.csv

Indentified robots file output: /auto/coucou/u/coucou/home/dtanasa/logs/found_bots.txt

Anonymization info store path: /auto/coucou/u/coucou/home/dtanasa/l

☑ Save result to the following databa: axis

○ Create new database ● Append to existing database axis_200302

☑ Analyze with Cluster & Discover ☐ Analyze with Divide & Discover

```
Joining logs ... done.
Removing requests from the Internal Robot ... done.
Removing requests for excluded extension files ... done.
Formating requests lines ... done.
Searching for WR with method robots.txt ... done.
```

Figure 4.8: Preprocessing Process: Running

4.1.2.4 Executing the Preprocessing Process

After entering all the information required in the application tabs, the preprocessing process can be started using the button *Run* from the toolbar or the *File->Run preprocessor* menu item.

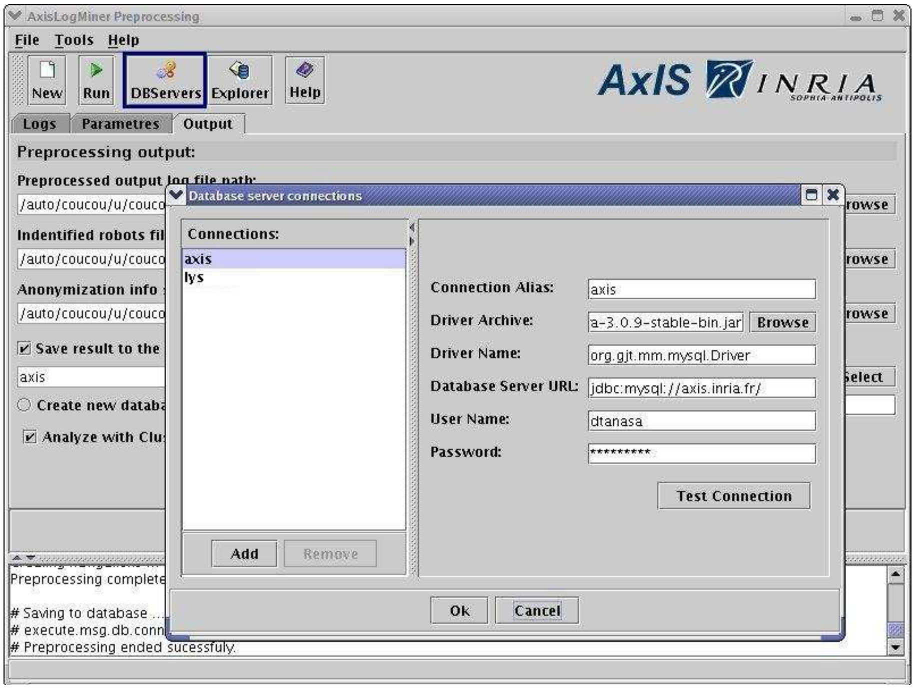
The process can be cancelled by the users at any moment by pressing the *Stop* button from the progress bar (see Figure 4.8). The preprocessing can take a long time, especially if there are multiple log files to be analyzed. However, as this is an off-line process, it can be scheduled to run over night in order not to interfere with other processes. Moreover, the database can be updated in an incremental manner, i.e. logs for the current month can be preprocessed and added to the database containing the sessions obtained from the previous periods analyzed.

4.1.3 Additional Tools

For managing the connections to the databases and for exploring the content of these databases, we created two additional tools integrated in the ALMP:

- **The Database Server Connections Manager** and
- **The Database Explorer**.

We illustrate in detail the two tools as follows.



Preprocessing output:

Preprocessed output log file path:
/auto/coucou/u/couco

Indentified robots fil
/auto/coucou/u/couco

Anonymization info
/auto/coucou/u/couco

☑ Save result to the
axis

○ Create new databa

☑ Analyze with Clu

Connections:
axis
lys

Connection Alias: axis
Driver Archive: a-3.0.9-stable-bin.jar
Driver Name: org.gjt.mm.mysql.Driver
Database Server URL: jdbc:mysql://axis.inria.fr/
User Name: dtanasa
Password: *********

```
Preprocessing complete

# Saving to database ...
# execute.msg.db.conn
# Preprocessing ended sucessfuly.
```

Figure 4.9: Database Server Connections Manager

4.1.3.1 Database Server Connections Manager

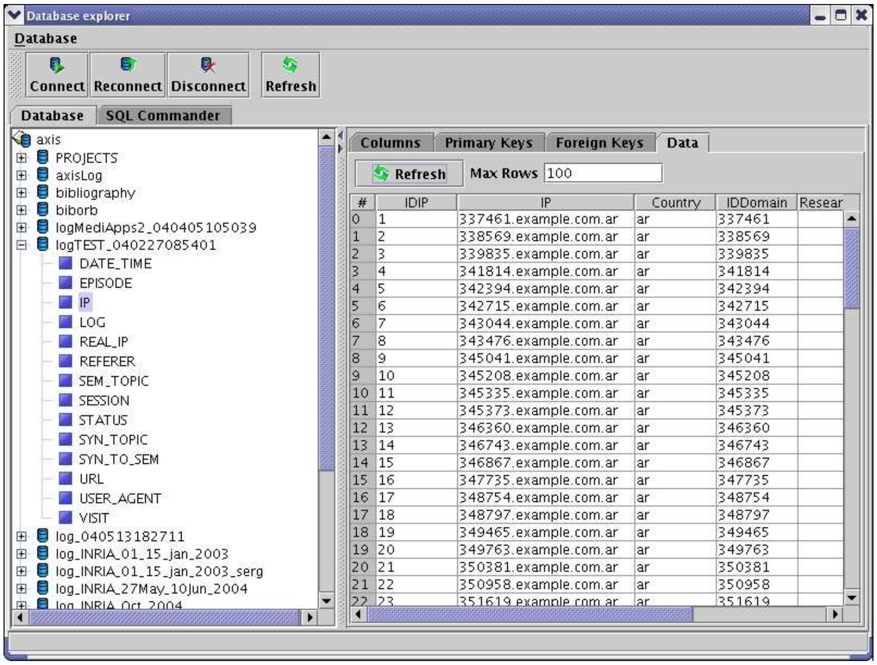
The Database Server Connections Manager (DBSCM) allows the users to manage the connections to the different database servers where they can store the preprocessed log files. To access this tool, the users can use the *DBServers* toolbar button (highlighted in Figure 4.9) or the *Tools->Database Servers* menu item.

The DBSCM window is shown in Figure 4.9. In the left hand side list are shown the different server connections while on the right hand side of the window, the database server connection information is entered/edited. The *Add* button must be used to create a new database server connection, the *Remove* button to delete an existing database server connection while the *Test Connection* button allows the testing of an existing database server connection.

4.1.3.2 Database Explorer

The Database Explorer tool (Figure 4.10) can be used to explore the structured log file stored in a database. The Database Explorer tool is available from the *Explorer* toolbar button or from the *Tools− >Database Explorer* menu item.

The connection to the database is controlled either by using the toolbar buttons or by using the menu items from the *Database* menu. To connect to a database server, the toolbar button *Connect* or the menu item with the same name must be used. This will show the connection manager we previously described.



axis
- PROJECTS
- axisLog
- bibliography
- biborb
- logMediApps2_040405105039
- logTEST_040227085401
  - DATE_TIME
  - EPISODE
  - IP
  - LOG
  - REAL_IP
  - REFERER
  - SEM_TOPIC
  - SESSION
  - STATUS
  - SYN_TOPIC
  - SYN_TO_SEM
  - URL
  - USER_AGENT
  - VISIT
- log_040513182711
- log_INRIA_01_15_jan_2003
- log_INRIA_01_15_jan_2003_serg
- log_INRIA_27May_10Jun_2004
- log_INRIA_Oct_2004

#	IDIP	IP	Country	IDDomain	Resear
0	1	337461.example.com.ar	ar	337461	
1	2	338569.example.com.ar	ar	338569	
2	3	339835.example.com.ar	ar	339835	
3	4	341814.example.com.ar	ar	341814	
4	5	342394.example.com.ar	ar	342394	
5	6	342715.example.com.ar	ar	342715	
6	7	343044.example.com.ar	ar	343044	
7	8	343476.example.com.ar	ar	343476	
8	9	345041.example.com.ar	ar	345041	
9	10	345208.example.com.ar	ar	345208	
10	11	345335.example.com.ar	ar	345335	
11	12	345373.example.com.ar	ar	345373	
12	13	346360.example.com.ar	ar	346360	
13	14	346743.example.com.ar	ar	346743	
14	15	346867.example.com.ar	ar	346867	
15	16	347735.example.com.ar	ar	347735	
16	17	348754.example.com.ar	ar	348754	
17	18	348797.example.com.ar	ar	348797	
18	19	349465.example.com.ar	ar	349465	
19	20	349763.example.com.ar	ar	349763	
20	21	350381.example.com.ar	ar	350381	
21	22	350958.example.com.ar	ar	350958	
22	23	351619.example.com.ar	ar	351619	

Figure 4.10: Database Explorer: Database Metadata and Data

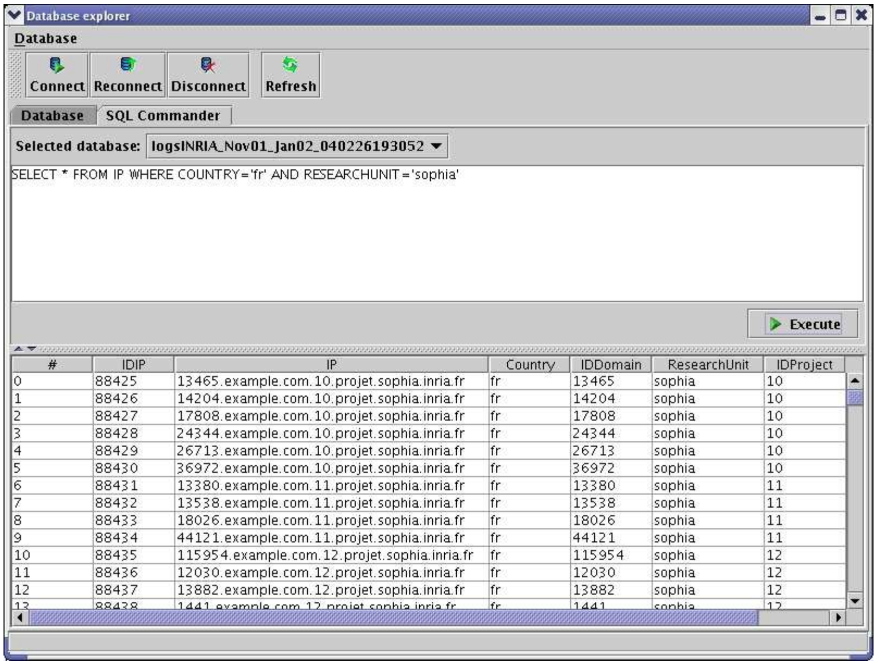
Once the database connection is established, the users can freely browse the database catalogs and tables using the left tree shown in the **Database** tab (see Figure 4.10). On the right hand side of the tree, the details for the current tree selection will be shown. For instance, in the case of a catalog selection in the left tree, the list of the catalog tables will be shown, on the right hand side.

When the user selects a table on the left hand side, information about the table primary keys, the table foreign keys, its columns and also the table data, are displayed in the right hand side tree (see Figure 4.10). To show these pieces of information, the user can chose the corresponding tab on the right hand side.

Finally, the **SQL Commander** tab (Figure 4.11) can be used to run user-defined SQL queries on the database or to perform custom data summaries.

4.2 Cluster & Discover Application

The Cluster & Discover (C&D) application (Figure 4.12) implements the method with the same name described in Chapter 3, Section 3.4. The graphical user interface of this application was designed in Java. We used Java for its cross-platform properties



Selected database: logsINRIA_Nov01_Jan02_040226193052

```
SELECT * FROM IP WHERE COUNTRY='fr' AND RESEARCHUNIT='sophia'
```

#	IDIP	IP	Country	IDDomain	ResearchUnit	IDProject
0	88425	13465.example.com.10.projet.sophia.inria.fr	fr	13465	sophia	10
1	88426	14204.example.com.10.projet.sophia.inria.fr	fr	14204	sophia	10
2	88427	17808.example.com.10.projet.sophia.inria.fr	fr	17808	sophia	10
3	88428	24344.example.com.10.projet.sophia.inria.fr	fr	24344	sophia	10
4	88429	26713.example.com.10.projet.sophia.inria.fr	fr	26713	sophia	10
5	88430	36972.example.com.10.projet.sophia.inria.fr	fr	36972	sophia	10
6	88431	13380.example.com.11.projet.sophia.inria.fr	fr	13380	sophia	11
7	88432	13538.example.com.11.projet.sophia.inria.fr	fr	13538	sophia	11
8	88433	18026.example.com.11.projet.sophia.inria.fr	fr	18026	sophia	11
9	88434	44121.example.com.11.projet.sophia.inria.fr	fr	44121	sophia	11
10	88435	115954.example.com.12.projet.sophia.inria.fr	fr	115954	sophia	12
11	88436	12030.example.com.12.projet.sophia.inria.fr	fr	12030	sophia	12
12	88437	13882.example.com.12.projet.sophia.inria.fr	fr	13882	sophia	12
13	88438	1441.example.com.12.projet.sophia.inria.fr	fr	1441	sophia	12

Figure 4.11: Database Explorer: SQL Commander

(C&D can run on both Linux and Windows platforms). However, some components of the application were developed in other languages either because they were previously implemented, like the SPM algorithm (PSP) developed in C, or because we used a language more appropriate for text file processing, like for the log file split script developed in Perl. We used Perl due to its performance in processing the text files.

In this section, we describe the user interface of the C&D application, the steps needed to analyze a structured log file using the method C&D and the output and result analysis for C&D.

4.2.1 Graphical User Interface

The main application window presented in Figure 4.12 contains:

1. The **menubar** and the corresponding **toolbar**. Using them, the user can start a new analysis (using the *New* button), open an existing analysis to continue it or to explore its results (using the *Open* button).

2. The **Log Division Panel**, situated on the left hand side under the toolbar, where the initial structured log file and the subsequent sub-logs are displayed.

3. The **Analyze** panels, located under the log division panel, are used to cluster, split, extract sequential patterns from a log/sub-log and to run the entire process. Their utilization will be further described in Section 4.2.2.

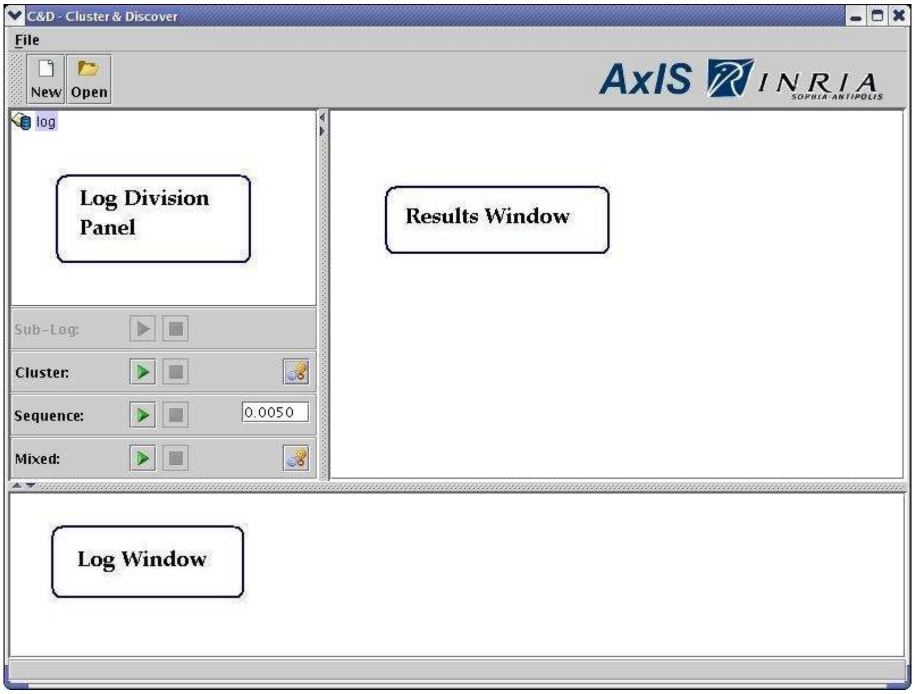


Figure 4.12: Cluster & Discover: New Analysis

4. The **Results Window**, placed on the right hand side of the main application window, displays the outputs from the clustering and SPM processes.

5. The **Log Window**, situated at the bottom of the main application window, displays the logging messages from different modules.

The GUI is intuitive and allows the user to perform all the basic operations from the analyze panels and to view the results of a new analysis (within the same application window).

4.2.2 New Analysis Using the C&D Application

To start a new analysis, the button *New* must be used and then the user can choose a structured log file from the files system. The application will create a new analysis folder, where all the temporary files will be stored. The structured log file will be represented on the log division panel (see Figure 4.12).

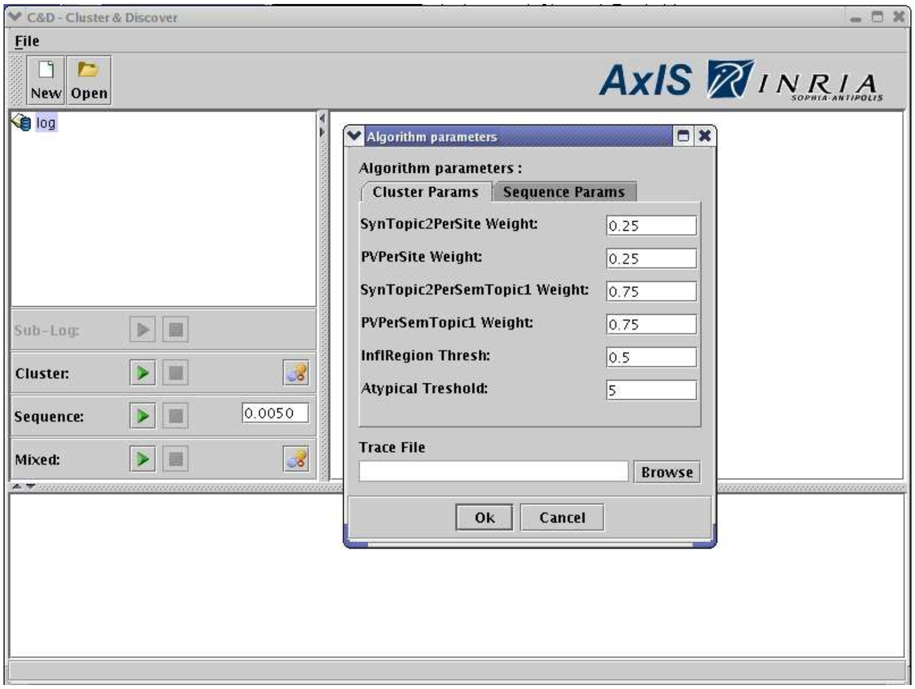
There are three ways of analyzing a structured log file:

1. Using the **Sequence** panel, the users can directly run an SPM analysis on the selected log file. The support value is entered in the textbox from this panel. The results of the SPM, if any, will be displayed on the Results Window while the output messages of the PSP program will be shown in the Log Window.

2. Using the **Cluster** panel, the users must set the clustering parameters first (using the *Settings* button on the Cluster panel) and then, they can launch the clustering process. At the end of the clustering process, if the results are satisfactory (i.e. number/size of clusters are in the expected range), the users can split the log by using the **Sub-log** panel, operation that will generate $n + 1$ sub-logs, n being the number of the clusters obtained (and there is one cluster that contains the sessions from the atypical clusters).

3. Using the **Mixed** panel, the users can directly launch the C&D process on the entire log. First, they will need to set the clustering and the SPM parameters using the Settings button from the Mixed panel. This will pop-up the *Algorithm parameters* window as shown in Figure 4.13. The users can enter the weight of the clustering algorithm (*Cluster params* tab) and the support range for the SPM algorithm (*Sequence Params* tab). Once the parameters are entered, they can launch the process by pressing on the Mixed button.

All four operational panels, situated under the Log Division Panel, contain one button for launching the process (on the left hand side) and one for cancelling (on the right hand side). The Cluster and Mixed panels also contain buttons for setting the parameters of the algorithms.



Sequence: 0.0050

Algorithm parameters :

SynTopic2PerSite Weight: 0.25

PVPerSite Weight: 0.25

SynTopic2PerSemTopic1 Weight: 0.75

PVPerSemTopic1 Weight: 0.75

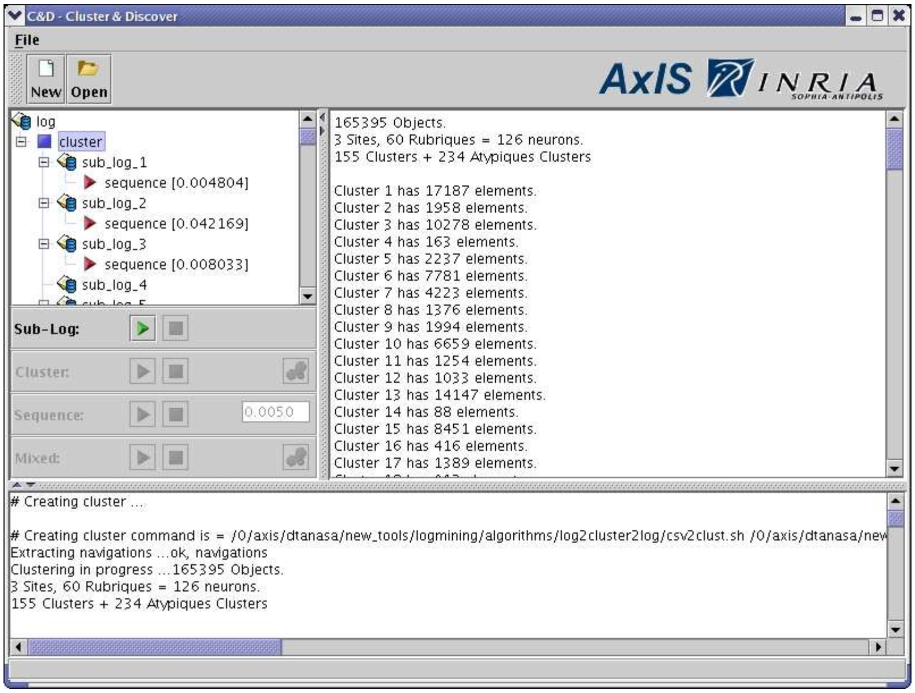
InflRegion Thresh: 0.5

Atypical Treshold: 5

Figure 4.13: Cluster & Discover: Parameters

4.2.3 Results Exploration

In the **Output window**, the user can display either the results of the SPM on the structured log or a sub-log, or the results of the clustering. In the latter case, the number of sessions in the log/sub-log is displayed together with the number of clusters and the number of sessions obtained for each cluster (see Figure 4.14).



```
165395 Objects.
3 Sites, 60 Rubriques = 126 neurons.
155 Clusters + 234 Atypiques Clusters

Cluster 1 has 17187 elements.
Cluster 2 has 1958 elements.
Cluster 3 has 10278 elements.
Cluster 4 has 163 elements.
Cluster 5 has 2237 elements.
Cluster 6 has 7781 elements.
Cluster 7 has 4223 elements.
Cluster 8 has 1376 elements.
Cluster 9 has 1994 elements.
Cluster 10 has 6659 elements.
Cluster 11 has 1254 elements.
Cluster 12 has 1033 elements.
Cluster 13 has 14147 elements.
Cluster 14 has 88 elements.
Cluster 15 has 8451 elements.
Cluster 16 has 416 elements.
Cluster 17 has 1389 elements.
```

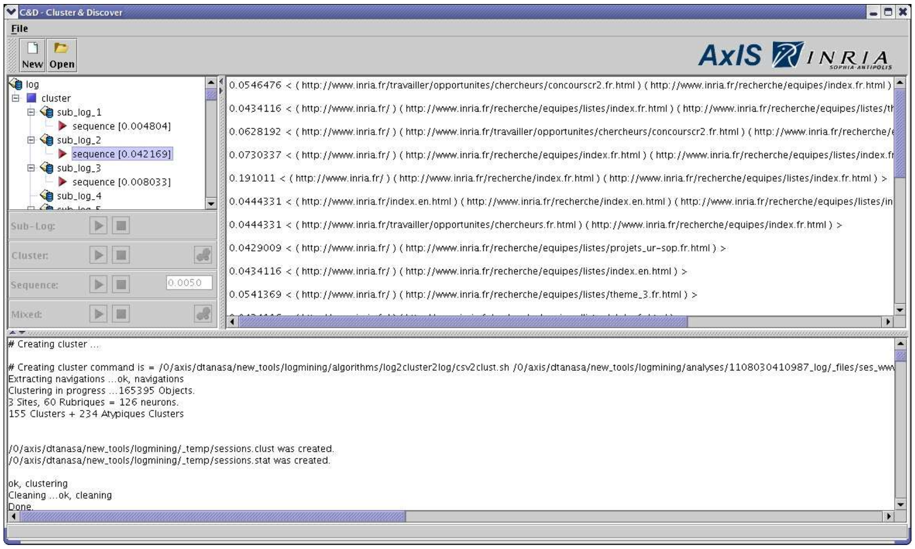
```
# Creating cluster ...

# Creating cluster command is = /0/axis/dtanasa/new_tools/logmining/algorithms/log2cluster2log/csv2clust.sh /0/axis/dtanasa/new
Extracting navigations ...ok, navigations
Clustering in progress ...165395 Objects.
3 Sites, 60 Rubriques = 126 neurons.
155 Clusters + 234 Atypiques Clusters
```

Figure 4.14: Cluster & Discover: Results

The output of the SPM algorithm consists in sequential patterns shown like in the Figure 4.15. For the sub-log selected, the user will see the corresponding support value and, for this value, all the sequential patterns extracted in the Output Window. The sequential patterns are listed one per line, with the support value next to each pattern. The user can search for a particular page in the Output Window. By right clicking on the window a popup menu with a *Find* command will appear.

In the “Mixed” mode, the users will have on the Log Division Panel, all the sub-logs generated for the initial log file and, for each sub-log, the sequential patterns extracted with the corresponding support. The value of the minimum support is calculated for each sub-log by using the global support value set by the user and the size of that sub-log. If the users want to analyze further a sub-log, they can run an SPM process at a different support value, or even launch a clustering or the C&D process on that sub-log (as described in the previous section).



```
0.0546476 < ( http://www.inria.fr/travailler/opportunites/chercheurs/concourscr2.fr.html ) ( http://www.inria.fr/recherche/equipes/index.fr.html )
0.0434116 < ( http://www.inria.fr/ ) ( http://www.inria.fr/recherche/equipes/listes/index.fr.html ) ( http://www.inria.fr/recherche/equipes/listes/th
0.0628192 < ( http://www.inria.fr/ ) ( http://www.inria.fr/travailler/opportunites/chercheurs/concourscr2.fr.html ) ( http://www.inria.fr/recherche/e
0.0730337 < ( http://www.inria.fr/ ) ( http://www.inria.fr/recherche/equipes/index.fr.html ) ( http://www.inria.fr/recherche/equipes/listes/index.fr
0.191011 < ( http://www.inria.fr/ ) ( http://www.inria.fr/recherche/index.fr.html ) ( http://www.inria.fr/recherche/equipes/listes/index.fr.html ) >
0.0444331 < ( http://www.inria.fr/index.en.html ) ( http://www.inria.fr/recherche/index.en.html ) ( http://www.inria.fr/recherche/equipes/listes/in
0.0444331 < ( http://www.inria.fr/travailler/opportunites/chercheurs.fr.html ) ( http://www.inria.fr/recherche/equipes/index.fr.html ) >
0.0429009 < ( http://www.inria.fr/ ) ( http://www.inria.fr/recherche/equipes/listes/projets_ur-sop.fr.html ) >
0.0434116 < ( http://www.inria.fr/ ) ( http://www.inria.fr/recherche/equipes/listes/index.en.html ) >
0.0541369 < ( http://www.inria.fr/ ) ( http://www.inria.fr/recherche/equipes/listes/theme_3.fr.html ) >
```

```
# Creating cluster ...

# Creating cluster command is = /0/axis/dtanasa/new_tools/logmining/algorithms/log2cluster2log/csv2clust.sh /0/axis/dtanasa/new_tools/logmining/analyses/1108030410987_log/_files/ses_www
Extracting navigations ...ok, navigations
Clustering in progress ...165395 Objects.
3 Sites, 60 Rubriques = 126 neurons.
155 Clusters + 234 Atypiques Clusters


/0/axis/dtanasa/new_tools/logmining/_temp/sessions.clust was created.
/0/axis/dtanasa/new_tools/logmining/_temp/sessions.stat was created.

ok, clustering
Cleaning ...ok, cleaning
Done.
```

Figure 4.15: Cluster & Discover: Results (Patterns)

4.3 Divide & Discover Application

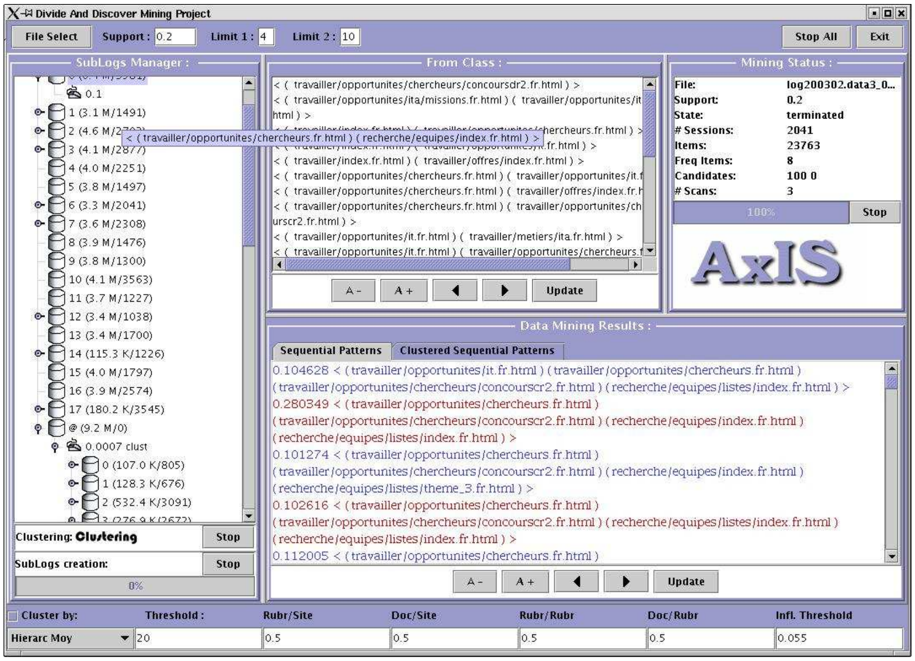
The Divide & Discover (D&D) tool was developed jointly with Florent Masseglia, previous to our C&D application. The GUI was developed in Java and, as the C&D application, it uses the PSP module (implemented in C).

The GUI is presented in Figure 4.16. Similar to the C&D application, we have the **Log Division Panel** (here it is called "Sublogs Manager") on the left hand side of the main window. The clustering and the SPM results are displayed in the **Data Mining Results Panel** and the content of a sub-log is shown in the panel above them. Parameters for the SPM and clustering can be entered on both the top and the bottom panels.

In Figure 4.16, we show the results of the SPM on one of the sub-logs. The output format is the same used in the C&D application.

4.4 Discussion and Future Work

The AxisLogMiner was designed as a useful WUM analysis toolbox that will assist the analyst in the three steps of the WUM process. The main advantage of using this toolbox over traditional log analyzing packages or commercial tools for WUM is the flexibility of AxisLogMiner. We designed our applications for WUM analysts, who will decide at each step, what data they need from the raw log files (in the AxisLogMiner Preprocessing) and, afterwards, what parameters should for the clustering and sequential mining process in order to find the most relevant patterns. We consider that,



Support : 0.2 Limit 1 : 4 Limit 2 : 10

```
0.1
1 (3.1 M/1491)
2 (4.6 M/2
3 (4.1 M/2877)
4 (4.0 M/2251)
5 (3.8 M/1497)
6 (3.3 M/2041)
7 (3.6 M/2308)
8 (3.9 M/1476)
9 (3.8 M/1300)
10 (4.1 M/3563)
11 (3.7 M/1227)
12 (3.4 M/1038)
13 (3.4 M/1700)
14 (115.3 K/1226)
15 (4.0 M/1797)
16 (3.9 M/2574)
17 (180.2 K/3545)
@ (9.2 M/0)
  0.0007 clust
    0 (107.0 K/805)
    1 (128.3 K/676)
    2 (532.4 K/3091)
```

Clustering: Clustering Stop
SubLogs creation: Stop
0%

```
< ( travailler/opportunites/chercheurs/concoursdr2.fr.html ) >
< ( travailler/opportunites/ita/missions.fr.html ) ( travailler/opportunites/it
html ) >
                                                                    hercheurs.fr.html ) >
< ( travailler/opportunites/chercheurs.fr.html ) ( recherche/equipes/index.fr.html ) >
                                                                    .fr.html ) >
< ( travailler/index.fr.html ) ( travailler/offres/index.fr.html ) >
< ( travailler/opportunites/chercheurs.fr.html ) ( travailler/opportunites/it.f
< ( travailler/opportunites/chercheurs.fr.html ) ( travailler/offres/index.fr.h
< ( travailler/opportunites/chercheurs.fr.html ) ( travailler/opportunites/ch
urscr2.fr.html ) >
< ( travailler/opportunites/it.fr.html ) ( travailler/metiers/ita.fr.html ) >
< ( travailler/opportunites/it.fr.html ) ( travailler/opportunites/chercheurs.f
```

File:	log200302.data3_0...
Support:	0.2
State:	terminated
# Sessions:	2041
Items:	23763
Freq Items:	8
Candidates:	100 0
# Scans:	3

100%

```
0.104628 < ( travailler/opportunites/it.fr.html ) ( travailler/opportunites/chercheurs.fr.html )
( travailler/opportunites/chercheurs/concourscr2.fr.html ) ( recherche/equipes/listes/index.fr.html ) >
0.280349 < ( travailler/opportunites/chercheurs.fr.html )
( travailler/opportunites/chercheurs/concourscr2.fr.html ) ( recherche/equipes/index.fr.html )
( recherche/equipes/listes/index.fr.html ) >
0.101274 < ( travailler/opportunites/chercheurs.fr.html )
( travailler/opportunites/chercheurs/concourscr2.fr.html ) ( recherche/equipes/index.fr.html )
( recherche/equipes/listes/theme_3.fr.html ) >
0.102616 < ( travailler/opportunites/chercheurs.fr.html )
( travailler/opportunites/chercheurs/concourscr2.fr.html ) ( recherche/equipes/listes/index.fr.html )
( recherche/equipes/listes/index.fr.html ) >
0.112005 < ( travailler/opportunites/chercheurs.fr.html )
```

□ Cluster by:	Threshold :	Rubr/Site	Doc/Site	Rubr/Rubr	Doc/Rubr	Infl. Threshold
Hierarc Moy	20	0.5	0.5	0.5	0.5	0.055

Figure 4.16: Divide & Discover: The Application

because of the diverse domain of WWW, this approach will allow the discovery of more interesting results than a more classical one, where the user can only choose between standard reports of users' activities.

Currently, we are in the process of integrating three other applications into the AxisLogMiner toolbox. The first is the Site Map [ST03] which captures and display the graph of a Web site. The second is the software tool corresponding to the Hierarchical Discovery method. Finally, we are designing a tool for assisting the user in the result exploration stage. For more details about these future extensions, please refer to 5.2.4.

Chapter 5

Conclusions and Perspectives

We conclude our thesis with a summary of our main contributions, some perspectives opened by our research and directions for future works.

5.1 Thesis Main Contributions

In this thesis, we presented a global approach for Web Usage Mining. As stated in Chapter 1, we designed a WUM approach that would overcome most of the existing open problems in WUM. Our global methodology for WUM offers solutions for preprocessing, data mining and a software toolbox that implements these methods.

The three main contributions of our thesis can be summarized as follows:

- We designed a general and complete methodology for data preprocessing in WUM that can also be used for preprocessing multiple Web log files in Intersites WUM;

- We proposed three approaches for extracting sequential patterns with low support from Web usage data: the Sequential Approach, the Iterative Approach and the Hierarchical Approach. These approaches were also instantiated in concrete methods such as the "Cluster & Discover" and "Divide & Discover".

- We designed and implemented a preprocessing tool supporting our methodology and two software tools for sequential pattern mining in WUM that follow the Sequential Approach (Cluster & Discover) and the Iterative Approach (Divide & Discover). All these tools were grouped together in the AxisLogMiner toolbox. The toolbox allows parameters passing between the applications and directly launching the SPM tools once the preprocessing process is finished.

5.1.1 A Complete Methodology for WUM

Our first contribution represents the general methodology for preprocessing described in Chapter 2. This methodology consists in four data preprocessing steps: data fusion, data cleaning, data structuration and data summarization. The first three steps are the classical preprocessing steps while the last one represents an advanced preprocessing step we created for summarizing and aggregating the structured data. Although some operations included in the various steps of this methodology were previously present in other works and especially in [Coo00], this is the first complete and general methodology for preprocessing in Intersites WUM.

Multiple log files coming from different Web servers are joined together in the **data fusion** step. Then, unnecessary requests are removed from this unique log file during the **data cleaning** step. The remaining requests are grouped in user sessions, visits and episodes and saved to a relational database designed according to our model presented in the **data structuration** step. In the last step of data preprocessing, the **data summarization**, summary variables are computed at different detail levels such as user session or visit. These variables are stored in the database and will be used in the data mining step that follows.

The preprocessing methodology was successfully tested on Web log files from INRIA's Web sites in the experiments presented in Section 2.7. Moreover, we preprocessed Web log files collected from INRIA[1] and UFPE[2] that were used afterwards by members of our team and partner teams in various clustering and statistical analysis. The preprocessing allowed a significant reduction in the size of the Web log files (down to 10 - 25% of the initial size) and also grouped requests in entities that could be further analyzed (visits, user sessions, semantic topics for URLs). Interesting results were obtained at all these levels and this emphasis the effectiveness of our methodology.

5.1.2 Three Approaches for Low Support SPM in WUM

We proposed a divisive methodology for sequential pattern mining in WUM in Chapter 3. This methodology aims to overcome problems that arise when applying classical SPM techniques on large Web logs. The classical SPM techniques are applied directly on the entire Web log file and, even when they are capable of returning results, these are either far too numerous (tens of thousands of sequential patterns) and cannot be used by the analyst or uninteresting and obvious results are obtained when the value of the minimum support is increased. To overcome these two problems we proposed three divisive approaches that combine both clustering and SPM techniques by:

1. Grouping similar sessions in clusters and, then
2. Extracting sequential patterns from these clusters through a classical SPM method.

The first approach proposed for this divisive methodology is the **Sequential Approach**. This approach combines a generic clustering algorithm applied on the entire structured log file with a generic SPM method applied on the results of the clustering. The support of the sequential patterns are recalculated in the final step with respect to the initial file.

[1] `www.inria.fr` and `www-sop.inria.fr`

[2] `www.cin.ufpe.br`

The second approach, the **Iterative Approach**, also uses a clustering algorithm and an SPM method. However, in this case, the clustering is applied on the sequential patterns resulted from an initial SPM. The log is split into non-disjunctive sub-logs corresponding to the clusters. The entire process can be repeated for each sub-log. This approach allows the analyst to “zoom” on groups of similar behaviors and, thus, facilitates the discovery of behaviors with very low support, corresponding to a minority group of users that made requests on a specific part of the Web site.

The third approach, the **Hierarchical Approach**, it is an iterative approach that uses the syntactic or semantic hierarchy of the Web pages. At each iteration, the hierarchical level for the Web pages is increased and, thus, more detail is available for a given page. The division of the log file is based on the sequential patterns discovered with a higher semantic/syntatic level for the Web pages. This ensures that sessions containing pages from the same category will be grouped in the same sub-log and, therefore, there are more chances that frequent minority behaviors are discovered by the SPM algorithm.

We instantiated the three approaches into three methods: the ”Cluster & Discover” method for the Sequential Approach, the ”Divide & Discover” method for the Iterative Approach and the ”Hierarchical Discover” method for the Hierarchical Approach. Using the software tools implemented for the first two methods, we extracted sequential patterns from the Web logs collected from two of INRIA’s Web sites. We compared the execution times obtained for the C&D method with the times obtained when executing two other SPM methods, the WAP-mine and PSP. For lower support values only the C&D method was capable of extracting sequential patterns (this is the case for three out of the eight sequential patterns presented in Section 3.7.3.1).

The sequential patterns discovered with the C&D and D&D methods represent precise and complex browsing behaviors that can be associated with a clear objective. They reveal various browsing interests, such as patterns about job opportunities, about specific software available for download on INRIA’s Web site, activities of the internal projects’ committee and lessons in a Java tutorial. We discovered also requests made by hacking tools that were trying to exploit known security holes from Web servers.

5.1.3 AxisLogMiner Toolbox

The AxisLogMiner was designed to be a toolbox for interlinked software applications, developed in Java, for Web Usage Mining. At the moment, three applications are included in this toolbox: the AxisLogMiner Preprocessing, the Cluster & Discover and the Divide & Discover. These tools allow the user to perform a complete WUM analysis, from the preprocessing of the Web log files until the result analysis step. Parameters can be passed between the three tools and the tools jointly use the database that we designed for storing the structured log file.

The **AxisLogMiner Preprocessing** implements the operations contained in our general methodology for preprocessing, detailed in Chapter 2. The users can choose to perform or skip some of these operations and they can also enter the parameters for the selected operations (e.g. extensions for the non-analyzed resources).

The **C&D** and **D&D** methods are also implemented as software applications using the Java language for the graphical user interface. These tools correspond to the Cluster & Discover method, and to the Divide & Discover method respectively. Using both tools, we were able to extract interesting and useful sequential patterns with support values as low as 0.009%.

5.2 Future Works and Perspectives

There are numerous perspectives opened by this research and they concern the three steps of the WUM as well as the AxisLogMiner Toolbox.

The future works and perspectives related to the WUM analysis, can be summarized by each WUM step, as follows:

- For **WUM preprocessing**: the preprocessing methodology presented in Chapter 2 can be further extended by using site maps for page view and episode identification; the site map could be considered with its different versions by using a *versionning* tool. Moreover, thanks to the extensible database model that takes into account the semantic topics for Web pages, the preprocessing methodology could be adapted to Web log files from the future Semantic Web sites.

- For **WUM pattern discovery**: new combinations of algorithms for the three proposed divisive approaches can be created; these new methods could afterwards be incorporated into the AxisLogMiner toolbox.

- For **WUM results analysis**: the most important perspective for this step would be the development of a tool that allows visual exploration of the patterns discovered; the Web site map can be used together with the SPM results to project them on the Web site structure; by using the semantic topics defined previously, the discrepancies between the Web masters' expectations and the WUM results could be highlighted.

The **AxisLogMiner toolbox** is conceived as a package of applications that can be easily extended by adding new software tools for the three steps of WUM. The first tool that we intend to add is the *Web Logic* application [ST03] that allows the extraction of the site map and its storage in a MySQL database or an XGMML file. Further extensions include the addition of result visualization tools and the improvement of its flexibility and existing linkages between the toolbox components.

Other topics that need to be investigated in the near future and not covered in this thesis are: the evaluation of patterns' quality, the need for better logging systems, the Semantic Web and its applications for WUM.

5.2.1 WUM Preprocessing

One of the most important future work for the preprocessing is the availability of the site structure (given by the site map). This can be used for the fulfilling some of the following WUM tasks:

- As mentioned in Section 2.5.3, the site map can be used for **identifying more precisely the page views** by determining which Web resources (e.g. images) are not contained in the Web pages, but can be displayed following explicit user request (i.e. by clicking on a link for the image).
- The **path completition** operation [Coo00] also needs the site map.
- The **episode identification** (Section 2.5.5) can be enhanced with a better site hierarchy built by coupling the syntactic topics (page physical location) and the site map (page logical location given by the hyperlinks).

However, the current structure of a Web site might not coincide with its structure at the time the logs were collected. Therefore, we need a mechanism to efficiently manage different versions of the site structure. For example, we can use XGMML source files, as in [PK01], and the Concurrent Versions System tool. The Web Logic tool that we developed is currently capable of saving the site map in the XGMML format [ST03].

5.2.2 Pattern Discovery Step

The site map could also be used in the SPM step, for instance, by the H&D method. This method needs the hierarchy of the syntactic topics to increase the level of detail at each iteration and, therefore, it could use a better site hierarchy based on the site map and, eventually, the syntactic topics.

The three approaches presented in Chapter 3 can be instantiated into new divisive methods using appropriate algorithms for clustering and sequential pattern mining. This will allow the user to compare the results of the different combinations of clustering and SPM techniques used for instantiating the approaches, in terms of the number of sequential patterns extracted as well as the quality (interestingness) of these patterns. We consider that a research on this issue will allow us to highlight the best clustering and SPM techniques to be used in a divisive approach for WUM. However, various datasets should be used and tested as the results obtained also depend on the Web site type, its structure and its users.

5.2.3 Result Analysis Step

As with other sequential mining techniques, an open problem of our methodology is represented by the quantity of the results obtained when using the methods proposed. Therefore, a future direction of our work is represented by the third step of WUM, the result analysis, on which we focused less in this research, compared with the two previous WUM steps.

A few tools for visualizing the results together with a representation of the Web site have already been developed in [KNY00, CSZG04, HW05]. But, a method for the result analysis step could also compare the existing Web structure with the sequential patterns discovered in order to highlight patterns involving ill-linked pages or to point out the need for new links.

5.2.4 Further Extensions to the AxisLogMining Toolbox

For the future work concerning the software development projects, we are currently in the process of including three additional applications in our toolbox:

- The WebLogic tool,
- The H&D software tool, and
- A new tool for exploring the sequential patterns obtained.

The **WebLogic** [ST03] is a fully developed application, although not yet included in our toolbox. It permits to capture and display the Web site map for any given Web site by using a graph structure (see Figure 5.1). It represents the Web pages and the links existing between them. The problems arising with this application are comparable with those encountered by similar applications, i.e. there is no standard way of graphically representing the Web site map. For some Web sites, neighbor pages may be placed in distant locations, or many edges (representing the links) may overlap or cross. We could solve these issues by offering several ways of displaying a Web site map (circular, tree, hierarchical tree, etc.).

The **H&D software tool** is curentlly under development and it implements the Hierarchical Discovery method. It is based on the Apriori-GST algorithm, already implemented for mining gene expression data (cf. Appendix B). The application's interface will be similar with C&D's GUI and will also include *Perl* scripts for splitting the log files based on the results of the sequential pattern mining.

The AxisLogMining toolbox contains tools for preprocessing and extracting sequential patterns from Web log files, but there is a need for more **advanced result analysis**. Therefore, we plan to develop a *Result Exploration* module that will be linked to the WebLogic application. This module will allow the user to select a sequential pattern extracted using one of the three applications (C&D, D&D or H&D) and to represent it on the Web site map.

Moreover, the flexibility of our toolbox should be increased so that the user could choose what methods to use for the three steps of the WUM. We imagine the AxisLogMiner as a modular toolbox, where the user will be able to select the appropriate methods/heuristics to be used in preprocessing and the aggregated variables to be filled in the database. Afterwards, the user can choose one of the three approaches for extracting the sequential patterns and, if needed, the algorithms that implement the approach. Finally, the results will be presented according to the objectives of the user analysis.

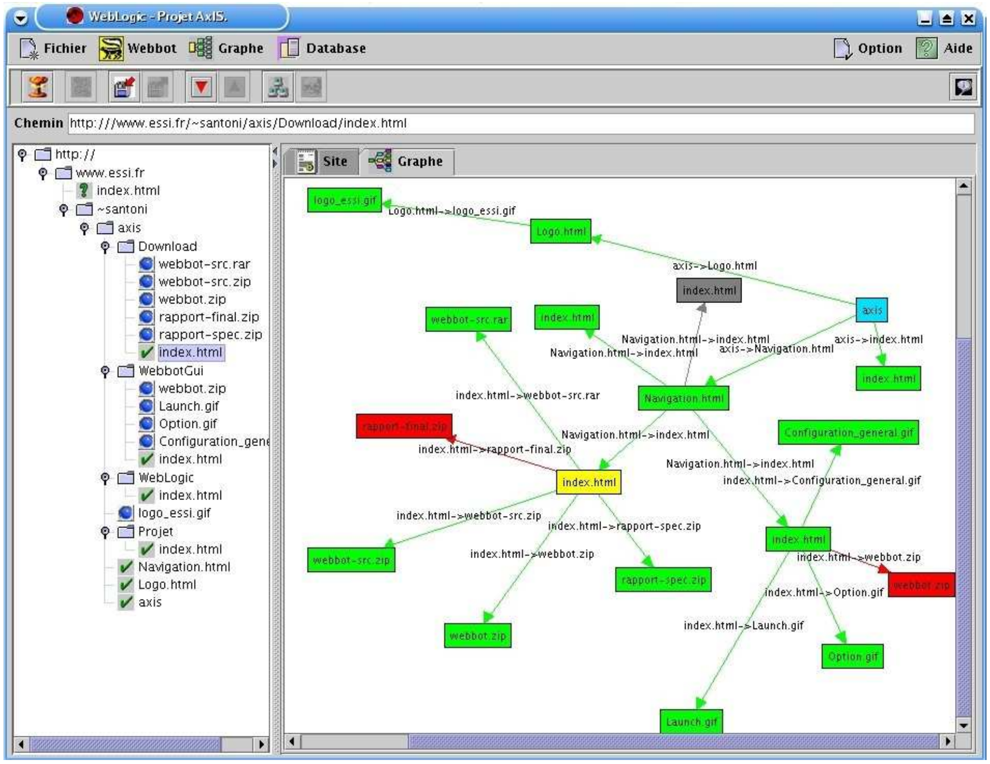


Figure 5.1: The WebLogic Application

5.2.5 Quality of SPM Results Assessment

As mentioned in Chapter 1, the main objective of our general methodology for sequential pattern mining in WUM was the improvement of the **quality of the patterns obtained**. We designed a divisive methodology in order to be able to extract sequential patterns with low support representing the coherent behaviors of small groups of users. As explained in [Gui04], finding these *nuggets* of information is more important than discovering frequent, but non surprising knowledge. There are few research works in WUM that discuss the topic of patterns' quality [HAC02, HWV03, Coo03].

We need to specify measures of interestingness and a methodology for SPM results assessment similar with those existing for association rules [Gui04]. A first measure could be represented by the confidence of a sequential pattern. As suggested in Chapter 1, we can compute the confidence for a sequential pattern (of length n) by aggregating the ratios between the sequential pattern's support and the support of the sequential patterns represented by a sub-sequence of $n-1$ items taken from the initial sequential pattern. Sequential patterns with a high confidence will have a higher *coherence* and, therefore, will be more interesting.

5.2.6 Improved Solutions for Logging Web Requests

An important issue of the Web Usage Mining is related to the ECLF log (or Combined log) format. This was designed in the early days of WWW, based on log files used in the Unix systems. At that time, the size of a Web site and the number of its visitors would have eventually allowed a manual examination for finding and fixing problems related to the Web site. But in the last 15 years, the Web grew exponentially and the need for better logging systems is acute. A new logging system allowing better WUM analysis would probably incorporate some of the steps from the preprocessing methodology presented. The database model we propose in Section 2.6 could be fed directly by such an advanced Web server logging system, by writing in the database the structured users' sessions, for example. This will lead to a better and errorless preprocessing and, therefore, to more accurate results in the SPM step. Moreover, the WUM analysis could even be executed at the same time that the data is collected (allowing a real time analysis).

5.2.7 Web Usage Mining and Semantic Web

The Semantic Web introduced in [BLF99] is a promising new domain. It proposes to represent the content published on the Web in a structured and standardized way so that it can be understood by an automated tool. The Semantic Web triggered propositions for a new range of Web Mining applications in [BHS02].

This standardized approach for representing the Web content will increase the quality of the Web pages, therefore, allowing better Web Mining results. With Web pages created in a structured language like XML, it is easier to extract information about topics presented in the pages and to associate specific content with a category (title, name, address, price, etc.). Therefore, Web Content Mining applications will be able to extract the necessary information from the Web pages easier and with less errors.

Moreover, with the Semantic Web, the links are explicitly described adding more information to the structure of the Web sites. This will allow better results for Web Structure Mining applications.

Finally, the Web Usage Mining can be improved by using, in the three steps of the WUM process, enriched information about the structure and content of the Web sites analyzed. Specific Web Usage Mining applications should be designed for mining the usage of Semantic Web sites. Their results will allow to enhance further the quality of the Web site and of the services that it provides.

Appendix A

List of Semantic Topics

ID	Semantic Topic	Description
1	accueil-siege	Homepage of `www.inria.fr`
2	accueil-sophia	Homepage of `www-sop.inria.fr`
3	actualites-siege	News page for `www.inria.fr`
4	actualites-sophia	News page for `www-sop.inria.fr`
5	agos-sophia	Social commission
6	bpas	Administration
7	cgi-bin	cgi scripts
8	challengeTV	“ChallengeTV” pages
9	colloquium	J. Moergenstern colloquiums
10	color	Local Research Collaborations
11	commun	Site map
12	ctime	Oracle Collaborative Suite pages
13	dias	Documentation center
14	didacticiel	Tutorials
15	direction	Research unit director
16	externe	External information
17	fonctions	Organizational chart
18	formation	Training
19	freesoft	Free software page
20	horde	Web mail page
21	icons	Icon images
22	intech	InTech group pages
23	international	International collaborations
24	interne-siege	`www.inria.fr` intranet
		Continued on the next page...

...continued from previous page		
ID	**Semantic Topic**	**Description**
25	interne-sophia	`www-sop.inria.fr` intranet
26	intro-inria	About us pages (Main site)
27	intro-sophia	About us pages (Sophia)
28	manifestations	Conferences
29	modeles	Templates for Web pages
30	multimedia	Multimedia resources
31	partenaires	Scientific partners
32	personnel	Staff pages
33	presse	Press releases
34	projets	Research teams
35	publications	Publications
36	ra	Activity reports
37	rapports	Technical and research reports
38	recherche	Research activity
39	rev	External relations service
40	rrrt	Technical reports
41	sapr	Projects' assistants service
42	scom	Communication service
43	semir	System administrators team
44	services	Services index
45	site-beta	Beta version of `www.inria.fr`
46	site-eng	English version of `www.inria.fr`
47	site-old	Old version of `www.inria.fr`
48	sophia	Pages about INRIA Sophia Antipolis
49	sophia_antipolis	Pages about Sophia Antipolis
50	thesauria	Research news
51	travailler	Careers
52	unites	Research units
53	valorisation	Valorization and transfer
54	w3c	W3C commitee
55	wiki	Wiki system pages

Table A.1: List of Semantic Topics for www.inria.fr and www-sop.inria.fr

A.1 Topics Correspondences for www.inria.fr

ID	Syntactic Topic	Semantic Topic
1	Accueil	accueil-siege
2	DR:	interne-siege
3	DR:I	interne-siege
4	DoctBase	formation
5	Formation	formation
6	Griseli	thesauria
7	JGI2001	manifestations
8	PICOF02	manifestations
9	PPSN2000	manifestations
10	RA95	ra
11	RA96	ra
12	RA97	ra
13	RA98	ra
14	RA99	ra
15	Ra-Dev96	ra
16	Ra-Dev97	ra
17	Ra-Dev98	ra
18	Ra-Dev99	ra
19	Ra-RI97	ra
20	Ra-Tech96	ra
21	Rapports	rapports
22	UCIS-AV+H	multimedia
23	Unites	unites
24	Watermarking	projets
25	acacia	projets
26	actualites	actualites-siege
27	agos-sophia	agos-sophia
28	aid	projets
29	ariana	projets
30	audiovisuel	multimedia
31	beta	site-beta
32	cafe	projets
33	caiman	projets
34	ce	travailler
35	cermics	partenaires
36	certilab	projets
37	cgi-bin	cgi-bin
38	chir	projets
39	cma	partenaires
40	comore	projets
41	croap	projets
42	dias	dias
43	dodo	formation
44	ecotel	manifestations
45	eg96	manifestations
46	epidaure	projets
47	fonctions	fonctions
48	freesoft	freesoft
49	icare	projets
50	icons	icons
51	inria	intro-inria
52	international	international
53	interne	interne-siege
54	koala	projets
55	lemme	projets
56	mascotte	projets
57	mefisto	projets
58	meije	projets
59	miaou	projets
60	mimosa	projets
61	mistral	projets
62	modeles	modeles
63	multimedia	multimedia
64	oasis	projets
65	odyssee	projets
66	old	site-old
67	omega	projets
68	opale	projets
69	orion	projets
70	pastis	projets
71	personnel	personnel
72	planete	projets
73	presse	presse
74	prisme	projets
75	publications	publications
76	rapports	rapports
77	rapportsactivite	ra
78	recherche	recherche
79	reves	projets
80	robotvis	projets
81	rodeo	projets
82	rrrt	rrrt
83	sadm	interne-siege
84	safir	projets
85	saga	projets
86	secoia	projets
87	semir	semir
88	sinus	projets
89	sloop	projets
90	sophia	sophia
91	sophia_antipolis	sophia_antipolis
92	stacs2002	manifestations
93	thesards	formation
94	thesauria	thesauria
95	tick	projets
96	travailler	travailler
97	tropics	projets
98	valorisation	valorisation
99	videos	multimedia
100	w3c	w3c

A.2 Topics Correspondences for www-sop.inria.fr

ID	Syntactic Topic	Semantic Topic
1	Accueil	accueil-sophia
2	COLOR	color
3	DR:	interne-sophia
4	DR:I	interne-sophia
5	EJC2000	manifestations
6	EJC2001	manifestations
7	EJC99	manifestations
8	Formation	formation
9	JGI2001	manifestations
10	LDTA2003	manifestations
11	Rapports	rapports
12	acacia	projets
13	aci	recherche
14	acigrid	recherche
15	act_recherche	recherche
16	actu	actualites-sophia
17	agos	agos-sophia
18	agos-sophia	agos-sophia
19	aid	projets
20	ariana	projets
21	axis	projets
22	bpas	bpas
23	cafe	projets
24	caiman	projets
25	cermics	partenaires
26	certilab	projets
27	cgi-bin	cgi-bin
28	challenge	challengeTV
29	chir	projects
30	cma	partenaires
31	colloquium	colloquium
32	commun	commun
33	comore	chir
34	concur98	manifestations
35	coprin	chir
36	croap	chir
37	ctime	ctime
38	dias	dias
39	didacticiel	didacticiel
40	direction	direction
41	dream	semir
42	eSmart2001	manifestations
43	ecotel	manifestations
44	eng	site-eng
45	epidaure	projets
46	express98	manifestations
47	externe	externe
48	freesoft	freesoft
49	galaad	projets
50	homepage	accueil-sophia
51	horde	horde
52	icare	projets
53	intech	intech
54	international	international
55	interne	interne-sophia
56	keystone	semir
57	koala	projets
58	lemme	projets
59	mascotte	projets
60	mefisto	projets
61	meije	projets
62	miaou	projets
63	mimosa	projets
64	mistral	projets
65	modeles	modeles
66	oasis	projets
67	odyssee	projets
68	omega	projets
69	opale	projets
70	orga_partenaires	partenaires
71	orion	projets
72	parallel	projets
73	planete	projets
74	presentation	intro-sophia
75	prisme	projets
76	rapports	rapports
77	relation_ext	rev
78	rev	rev
79	reves	projets
80	robotvis	projets
81	rodeo	projets
82	safir	projets
83	saga	projets
84	sapr	sapr
85	scom	scom
86	secoia	projets
87	semir	semir
88	services	services
89	sinus	projets
90	sloop	projets
91	smash	projets
92	sophia	sophia
93	stacs2002	manifestations
94	sysdys	projets
95	thesards	formation
96	tick	projets
97	tropics	projets
98	visa	projets
99	w3c	partenaires
100	wiki	wiki

Appendix B

Using Apriori-GST to Extract Sequential Patterns for Gene Regulatory Expressions Profiles

An extended version of this study, describing an application of the Apriori-GST algorithm on microarray data, was presented at KELSI 2004, Milano, Italy [TLT04].

B.1 Background

Given the advent of microarray technology, it is now possible to analyze the expression of a large number of genes simultaneously [LFGL99]. Microarray experiments can be classified according to the nature of the samples, i.e. time of collection, location, type of tissue, class of tumor, etc. In this appendix, we explore the results obtained by applying our Apriori-GST algorithm to time series microarray experiments. In particular, we report results applied to gene expression time series associated to Mouse Cerebellum development [KF01, LWP00, WDM$^+$97].

B.1.1 Biological Motivation and Gene Expression Data Generation

In this appendix, we present outcomes of our computational analysis applied to time-series gene expression data generated by Kagami et al [KF01]. This data is publicly available through the Gene Expression Omnibus [Omn]. In this study, Kagami et al [KF01] investigated differentially expressed genes during the development of mouse cerebellum. Their biological interest was focused to further understand the molecular basis of mouse cerebellum development. The mouse cerebellum is not entirely developed until post-natal day 21, therefore, their experiment was an ideal framework for the understanding of the genetic foundations and the mechanisms of neural development [MKS$^+$00].

Gene expression data was generated using Affimetrix chips and validated by quantitative RT-PCR expression analysis of 12 randomly selected and differentially expressed genes. Samples were independently hybridized in duplicate with 25-mer oligonucleotide sequences representing 12,654 genes on Mu11K GeneChips.

The biotin-labelled cRNA samples generated from mouse cerebella samples were collected at five developmental stages, 1 prenatal (embryonic day 18 or E18) and 4 postnatal at 7, 14, 21 and 56 days (P7, P14, P21 and P56). The postnatal morphological and neurological development of the mouse cerebellum was studied and reported [Alt72, GH98].

B.1.2 Suffix Trees and Gene Data

Suffix Trees are widely used in computational biology, especially for genome alignment [Gus97, LL03]. As far as we know, our approach of using an algorithm based on Generalized Suffix Trees (GST) and Apriori for microarray data is an original approach for this domain. Previously, we successfully applied our Apriori-GST algorithm to Web Usage data and the results are reported in [TT01]. The Apriori-GST algorithm is described in Section 3.6.1.

B.2 Method

As previously described in Chapter 3, sequential pattern mining is an advanced data mining technique that extracts frequently occurring patterns in given sequences. As an example, from a dataset for market basket analysis, one could find a pattern like "*Customers who buy digital cameras will later buy memory cards and then photo printers*".

With a proper preprocessing, these kind of patterns can also be extracted from microarray data. We use the Apriori-GST algorithm to extract contiguous sequential patterns from the preprocessed micro-array data.

The microarray data is transformed into sequences of three possible levels of exposure (e^+, e^0 or e^-), cf. Section B.2.1. These sequences are indexed using a GST index (cf. Section 3.6.1.2). A microarray sequential pattern may be seen, in this case, as a subsequence of levels of exposures that frequently occur. We will apply the Apriori-GST algorithm, as described below, to discover such sequences.

B.2.1 Encoding Microarray Data

The original dataset taken from [KF01] contains 5 series of normalized expressed values for 897 genes. We consider this dataset as 897 sequences (time ordered), so we have one sequence for each gene G_i.

Thus, we can express a gene as:

$$G_i =< G_i(t_1),\ G_i(t_2),\ G_i(t_3),\ G_i(t_4),\ G_i(t_5) >,$$

each $G_i(t_j)$ representing the normalized expression value for gene G_i at time t_j.

Next, we replace the normalized values with their natural logarithm values, obtaining:

$$Log(Gi) =< log(G_i(t_1)),\ log(G_i(t_2))\ ...\ log(G_i(t_5)) >.$$

We compute the mean ($Mean$) and the standard deviation (STD) for the gene expression values of the initial 5 series:

$$Mean(t_j) = Mean(log(G_i(t_j))), \text{ for each } i = \overline{1..897}, \text{ for each } j = \overline{1..5}$$

$$STD(t_j) = STD(log(G_i(t_j))), \text{ for each } i = \overline{1..897}, \text{ for each } j = \overline{1..5}$$

We compute the $ZScore$ as follows:

$$ZScore(G_i(t_j)) = \frac{log(G_i(t_j)) - Mean(t_j)}{STD(t_j)}, \text{ for each } i = \overline{1..897}, \text{ for each } j = \overline{1..5}$$

Finally, based on its $ZScore$, to each $G_i(t_j)$ is given one of the three values for the level of exposure (e^+ - *expressed at high level*, e^0 - *expressed at medium level* or e^- - *expressed at low level*) according to the following algorithm:

```
if $ZScore(G_i(t_j)) \geq 1.96$ then $G_i(t_j) = e^+$
else if $ZScore(G_i(t_j)) \leq -1.96$ then $G_i(t_j) = e^-$
else $G_i(t_j) = e^0$.
```

Thus, at the end of the preprocessing step, the dataset contains sequences of items with three possible values.

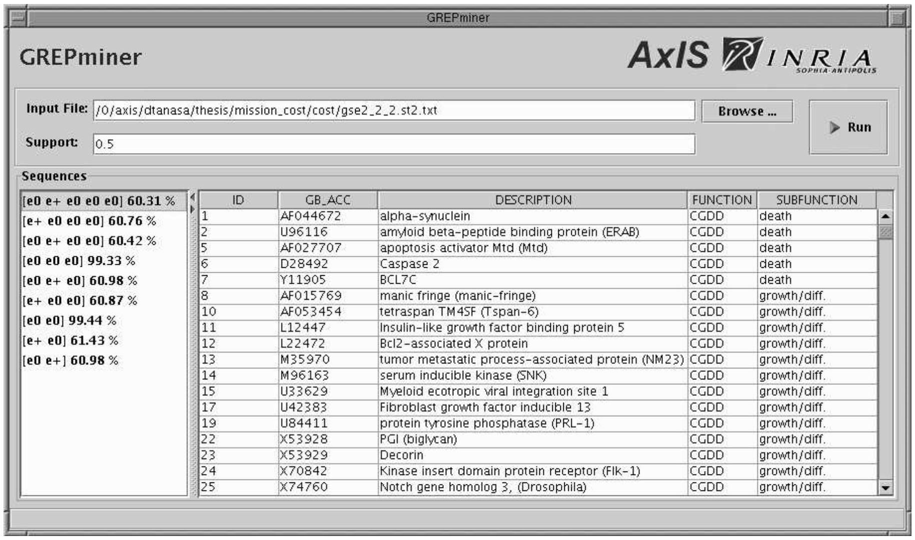
B.2.2 Indexing Gene Regulatory Expressions

In order to index the gene regulatory expressions (GREs), we use the GST index [Gus97] described in Section 3.6.1.2. The GST index will be further used by the Apriori-GST algorithm as described in Section 3.6.1.

B.3 Results

To support our methodology, we designed and implemented in Java, the GREPminer[1] tool presented in Figure B.1. The user chooses a preprocessed file as described in Section B.2.1 and extracts sequential patterns having the support superior to a specified threshold. The frequent sequential patterns extracted are listed on the left side and details (list of genes) for the selected pattern are displayed on the right side.

[1] Gene Regulatory Expression Profiles Miner



Input File: /0/axis/dtanasa/thesis/mission_cost/cost/gse2_2_2.st2.txt

Support: 0.5

[e0 e+ e0 e0 e0] 60.31 %
[e+ e0 e0 e0] 60.76 %
[e0 e+ e0 e0] 60.42 %
[e0 e0 e0] 99.33 %
[e0 e+ e0] 60.98 %
[e+ e0 e0] 60.87 %
[e0 e0] 99.44 %
[e+ e0] 61.43 %
[e0 e+] 60.98 %

ID	GB_ACC	DESCRIPTION	FUNCTION	SUBFUNCTION
1	AF044672	alpha-synuclein	CGDD	death
2	U96116	amyloid beta-peptide binding protein (ERAB)	CGDD	death
5	AF027707	apoptosis activator Mtd (Mtd)	CGDD	death
6	D28492	Caspase 2	CGDD	death
7	Y11905	BCL7C	CGDD	death
8	AF015769	manic fringe (manic-fringe)	CGDD	growth/diff.
10	AF053454	tetraspan TM4SF (Tspan-6)	CGDD	growth/diff.
11	L12447	Insulin-like growth factor binding protein 5	CGDD	growth/diff.
12	L22472	Bcl2-associated X protein	CGDD	growth/diff.
13	M35970	tumor metastatic process-associated protein (NM23)	CGDD	growth/diff.
14	M96163	serum inducible kinase (SNK)	CGDD	growth/diff.
15	U33629	Myeloid ecotropic viral integration site 1	CGDD	growth/diff.
17	U42383	Fibroblast growth factor inducible 13	CGDD	growth/diff.
19	U84411	protein tyrosine phosphatase (PRL-1)	CGDD	growth/diff.
22	X53928	PGI (biglycan)	CGDD	growth/diff.
23	X53929	Decorin	CGDD	growth/diff.
24	X70842	Kinase insert domain protein receptor (Flk-1)	CGDD	growth/diff.
25	X74760	Notch gene homolog 3, (Drosophila)	CGDD	growth/diff.

Figure B.1: The GREPminer Tool Implementing the Apriori-GST Algorithm

We used a temporal dataset described in [KF01] taken from the GEO repository. The dataset consisted in 5 series of expression levels for 897 genes. The gene list comprises only those genes that Kagami et al. [KF01] have selected as being significantly expressed. Their gene-selection criteria were:

- Show more than a two-fold difference between maximum and minimum intensity,
- The maximum intensity must be > 1000 units, and
- The differential expression must be seen in both replicates.

We executed several tests. First, we extracted frequent sequential patterns having the support superior to 50% (cf. Table B.3). Next, we extracted all the patterns from this dataset by specifying a support of at least 1 sequence (*i.e.* 0.11%) and we obtained 40 patterns as listed in Table B.3.

As this was a small dataset, the execution time for our application was less than 1 second for the lowest possible support for this dataset (0.11%).

B.4 Discussion and Conclusions

In Table B.3, the interesting pattern is SP_1 as it concerns a large number of genes (60.31%) and consists in a high expressed gene on the postnatal P7 and medium expressed on the other stages. This leads us to the conclusion that there is a lot of gene activity around the stage P7.

The Table B.4, containing all the patterns of length 5, can be regarded as a partition over the whole set of the genes studied. Here, a number of interesting time-patterns can

PatternID	Pattern	Support	Number of Genes
SP_1	$[e^0\ e^+\ e^0\ e^0\ e^0]$	60.31%	541
SP_2	$[e^+\ e^0\ e^0\ e^0]$	60.76%	545
SP_3	$[e^0\ e^+\ e^0\ e^0]$	60.42%	542
SP_4	$[e^0\ e^0\ e^0]$	99.33%	891
SP_5	$[e^0\ e^+\ e^0]$	60.98%	547
SP_6	$[e^+\ e^0\ e^0]$	60.87%	546
SP_7	$[e^0\ e^0]$	99.44%	892
SP_8	$[e^+\ e^0]$	61.43%	551
SP_9	$[e^0\ e^+]$	60.98%	547

Table B.1: List of All Patterns with Support > 50%

be observed. The first two patterns, SP_1 ($[e^0\ e^+\ e^0\ e^0\ e^0]$) and SP_2 ($[e^0\ e^-\ e^0\ e^0\ e^0]$) are associated with high support figures, 60.31% and 16.05% respectively.

The first pattern, which is the most supported, proves that a high number of genes are highly expressed at the first postnatal observed stage (P7), but medium expressed otherwise. In contrast, the second pattern represents genes that from medium expressed values and go to low expressed values on P7. In fact, excepting two patterns (SP_4 and SP_9), all the other patterns change their expression values on or after the postnatal stage P7. Together, all these patterns represent 90.63% of the entire genes list.

These genes change their expressed value around stage P7 and, except for a minority of 6 genes supporting SP_6, SP_7 and SP_{10}, they keep a medium expressed value for the rest of the postnatal stages. Our hypothesis is that there is a lot of gene activity between the prenatal stage E18 and postnatal stage P7 and the authors of the initial study [KF01] disregarded the intermediary stages, between E18 and P7 (*i.e.* P0 and P3). Our belief is that a more detailed study, including these two stages and also stages between P7 and P14, would allow a better classification of the genes.

The pattern SP_4 is not of much interest as the 83 genes supporting this pattern have a medium expressed value during the 5 stages observed.

The rest of the patterns (from SP_5 to SP_{10}) represent a small number of genes or even single genes like SP_9 and SP_{10}.

The patterns SP_6 and SP_7 are supported by genes with highly similar behavior. The difference between the two groups resides in the stage P56 when genes from SP_6 are expressed at low level and genes from SP_7 are expressed at medium level.

The two genes supporting SP_8 are expressed at low level in the prenatal stage and then expressed at medium level for the rest of the stages (highly similar with SP_4).

Both patterns, SP_9 and SP_{10}, are supported by a single gene, the gene 623, and 671

PatternID	Pattern	Support	Number of Genes
SP_1	$[e^0\ e^+\ e^0\ e^0\ e^0]$	60.31	541
SP_2	$[e^0\ e^-\ e^0\ e^0\ e^0]$	16.05	144
SP_3	$[e^-\ e^-\ e^0\ e^0\ e^0]$	12.93	116
SP_4	$[e^0\ e^0\ e^0\ e^0\ e^0]$	9.25	83
SP_5	$[e^-\ e^+\ e^0\ e^0\ e^0]$	0.45	4
SP_6	$[e^0\ e^+\ e^0\ e^-\ e^-]$	0.33	3
SP_7	$[e^0\ e^+\ e^0\ e^-\ e^0]$	0.22	2
SP_8	$[e^-\ e^0\ e^0\ e^0\ e^0]$	0.22	2
SP_9	$[e^0\ e^0\ e^0\ e^-\ e^0]$	0.11	1
SP_{10}	$[e^0\ e^+\ e^0\ e^0\ e^-]$	0.11	1
SP_{11}	$[e^+\ e^0\ e^0\ e^0]$	60.76	545
SP_{12}	$[e^0\ e^+\ e^0\ e^0]$	60.42	542
SP_{13}	$[e^-\ e^0\ e^0\ e^0]$	29.21	262
SP_{14}	$[e^0\ e^-\ e^0\ e^0]$	16.05	144
SP_{15}	$[e^-\ e^-\ e^0\ e^0]$	12.93	116
SP_{16}	$[e^0\ e^0\ e^0\ e^0]$	9.48	85
SP_{17}	$[e^0\ e^+\ e^0\ e^-]$	0.56	5
SP_{18}	$[e^-\ e^+\ e^0\ e^0]$	0.45	4
SP_{19}	$[e^+\ e^0\ e^-\ e^-]$	0.33	3
SP_{20}	$[e^+\ e^0\ e^-\ e^0]$	0.22	2
SP_{21}	$[e^+\ e^0\ e^0\ e^-]$	0.11	1
SP_{22}	$[e^0\ e^0\ e^0\ e^-]$	0.11	1
SP_{23}	$[e^0\ e^0\ e^-\ e^0]$	0.11	1
SP_{24}	$[e^0\ e^0\ e^0]$	99.33	891
SP_{25}	$[e^0\ e^+\ e^0]$	60.98	547
SP_{26}	$[e^+\ e^0\ e^0]$	60.87	546
SP_{27}	$[e^-\ e^0\ e^0]$	29.21	262
SP_{28}	$[e^0\ e^-\ e^0]$	16.39	147
SP_{29}	$[e^-\ e^-\ e^0]$	12.93	116
SP_{30}	$[e^+\ e^0\ e^-]$	0.56	5
SP_{31}	$[e^-\ e^+\ e^0]$	0.45	4
SP_{32}	$[e^0\ e^-\ e^-]$	0.33	3
SP_{33}	$[e^0\ e^0\ e^-]$	0.22	2
SP_{34}	$[e^0\ e^0]$	99.44	892
SP_{35}	$[e^+\ e^0]$	61.43	551
SP_{36}	$[e^0\ e^+]$	60.98	547
SP_{37}	$[e^-\ e^0]$	29.54	265
SP_{38}	$[e^0\ e^-]$	16.83	151
SP_{39}	$[e^-\ e^-]$	13.27	119
SP_{40}	$[e^-\ e^+]$	0.45	4

Table B.2: List of All Patterns Extracted

PatternID	Pattern	Support	Number of Genes
SP_1	$[e^0\ e^+\ e^0\ e^0\ e^0]$	60.31	541
SP_2	$[e^0\ e^-\ e^0\ e^0\ e^0]$	16.05	144
SP_3	$[e^-\ e^-\ e^0\ e^0\ e^0]$	12.93	116
SP_4	$[e^0\ e^0\ e^0\ e^0\ e^0]$	9.25	83
SP_5	$[e^-\ e^+\ e^0\ e^0\ e^0]$	0.45	4
SP_6	$[e^0\ e^+\ e^0\ e^-\ e^-]$	0.33	3
SP_7	$[e^0\ e^+\ e^0\ e^-\ e^0]$	0.22	2
SP_8	$[e^-\ e^0\ e^0\ e^0\ e^0]$	0.22	2
SP_9	$[e^0\ e^0\ e^0\ e^-\ e^0]$	0.11	1
SP_{10}	$[e^0\ e^+\ e^0\ e^0\ e^-]$	0.11	1

Table B.3: List of All Patterns of Length 5 (Partition over the Dataset)

PatternID	ID	GB_ACC	DESCRIPTION	FUNCTION	SUBFUNCTION
SP_9	623	D45208	HPC-1/syntaxin	IMTN	vesicle
SP_{10}	671	X67668	High mobility group protein 2	NNM	

Table B.4: Details of the 2 Singular Genes

respectively. We can affirm that these genes have an “uncommon” behavior. The description for the two genes is given in Table B.4.

The pattern SP_5 is the only pattern containing a stage at low expressed value followed by a stage at high expressed value. We included in Table B.5 the details of this pattern.

Through this study we highlighted the essential stages for genes activity in Mouse Cerebellum development, but these stages need further investigation in a more precise study for more detailed results.

ID	GB_ACC	DESCRIPTION	FUNCTION	SUBFUNCTION
178	X02801	Glial fibrillary acidic protein (GFAP)	CSC	cytoskeleton
462	AI838274	3’ end /clone=UI-M-AO0-aby-a-05-0-UI	EST	
619	U48398	Aquaporin 4	IMTN	transporter
896	X13986	minopontin	UC	

Table B.5: Details on the 4 Genes Supporting the SP_5 Pattern

Bibliography

[ABC] ABCInteactive.com. Spiders and Robots. http://www.abcinteractiveaudits.com/abci_iab_spidersandrobots/.

[AG91] A. Azcarraza and A. Giacometti. A Prototype-Based Incremental Network Model for Classification Task. In *Proceedings of the Forth International Conference on Neural Networks and Their Applications*, pages 121–134, Nimes, France, 1991.

[AIS93] R. Agrawal, T. Imielinski, and A. Swami. Mining Association Rules between Sets of Items in Large Databases. In *Proc. of the 1993 ACM SIGMOD Conference*, pages 207–216, Washington DC, USA, May 1993.

[Alt72] J. Altman. Postnatal Development of the Cerebellar Cortex in the Rat. *Journal of Comparative Neurology*, 145(3-4):353–513, 1972.

[ALT$^+$03] M. Arnoux, Y. Lechevallier, D. Tanasa, B. Trousse, and R. Verde. Automatic Clustering for the Web Usage Mining. In D. Petcu, D. Zaharie, V. Negru, and T. Jebeleanu, editors, *Proceedings of the Fifth International Workshop on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC03)*, pages 54–66. Editura Mirton, Timisoara, 1-4 October 2003.

[ALT$^+$04] M. Arnoux, Y. Lechevallier, D. Tanasa, B. Trousse, and R. Verde. Classification automatique à partir de logs web et de connaissances sur le site. In *Atelier "Fouille de Données Complexes" à EGC'04*, pages 59–72, Clermont-Ferrand, January 2004.

[AS94] R. Agrawal and R. Srikant. Fast Algorithms for Mining Association Rules. In Jorge B. Bocca, Matthias Jarke, and Carlo Zaniolo, editors, *Proceedings of the Twentieth International Conference on Very Large Data Bases, VLDB*, pages 487–499. Morgan Kaufmann, September 1994.

[AS95] R. Agrawal and R. Srikant. Mining Sequential Patterns. In P.S. Yu and A.L. P. Chen, editors, *Proceedings of the Eleventh International Conference on Data Engineering, March 6-10, 1995, Taipei, Taiwan*, pages 3–14. IEEE Computer Society, 1995.

[BAC04] P. De Bra, L. Aroyo, and V. Chepegin. The Next Big Thing: Adaptive Web-Based Systems. *Journal of Digital Information*, 5(1), May 2004. Article no 247, `http://jodi.ecs.soton.ac.uk/Articles/v05/i01/DeBra/`.

[BGG$^+$01] F. Bonchi, F. Giannotti, C. Gozzi, G. Manco, M. Nanni, D. Pedreschi, C. Renso, and S. Ruggieri. Web Log Data Warehousing and Mining for Intelligent Web Caching. *Data Knowledge Engineering*, 39(2):165–189, 2001.

[BGR01] V. Berry, O. Gascuel, and E. Rivals. Méthodes et Algorithmes pour la Génomique (Cours DEA Informatique). LIRMM Montpellier, 2001. `http://www.lirmm.fr/~rivals/DEA/`.

[BHS02] B. Berendt, A. Hotho, and G. Stumme. Towards Semantic Web Mining. In *Proceedings of the First International Semantic Web Conference on The Semantic Web (ISWC'02)*, pages 264–278. Springer-Verlag, 2002.

[BLF99] T. Berners-Lee and M. Fischetti. *Weaving the Web*. HarperCollins, New York, NY, 1999.

[BLFM98] T. Berners-Lee, R. Fielding, and L. Masinter. Uniform Resource Identifiers (URI): Generic Syntax. Network Working Group RFC 2396, `http://www.rfc-editor.org/rfc/rfc2396.txt`, August 1998.

[BMNS02] B. Berendt, B. Mobasher, M. Nakagawa, and M. Spiliopoulou. The Impact of Site Structure and User Environment on Session Reconstruction in Web Usage Analysis. In *Proceedings of the Forth WebKDD 2002 Workshop, at the ACM-SIGKDD Conference on Knowledge Discovery in Databases (KDD'2002)*, Edmonton, Alberta, Canada, July 2002.

[BMSW01] B. Berendt, B. Mobasher, M. Spiliopoulou, and J. Wiltshire. Measuring the Accuracy of Sessionizers for Web Usage Analysis. In *Proceedings of the Workshop on Web Mining at the First SIAM International Conference on Data Mining*, 2001.

[BS00] B. Berendt and M. Spiliopoulou. Analysis of Navigation Behaviour in Web Sites Integrating Multiple Information Systems. *VLDB*, 9(1):56–75, 2000.

[BT02] A. Benedek and B. Trousse. Adaptation of Self-Organizing Maps for CBR Case Indexing. In *Proceedings of the Forth International Workshop on Symbolic and Numeric Algorithms for Scientific Computing*, pages 31–45, Timisoara, Romania, October 2002. and also 27th Annual Conference of the Gesellschaft fur Klassifikation, 12–14 March 2003, Cottbus, Germany.

[CBT04] S. Chelcea, P. Bertrand, and B. Trousse. A New Agglomerative 2-3 Hierarchical Clustering Algorithm. In *Innovations in Classification,* *Data Science, and Information Systems. Proceedings of 27^{t}h Annual GfKl Conference, University of Cottbus, March 12 - 14, 2003*, Studies in Classification, Data Analysis, and Knowledge Organization, pages 3–10, Heidelberg-Berlin, Germany, 2004. Springer-Verlag.

[CHM$^+$00] I. V. Cadez, D. Heckerman, C. Meek, P. Smyth, and S. White. Visualization of Navigation Patterns on a Web Site Using Model-Based Clustering. In *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 280–284, Boston, Massachusetts, 2000.

[CHS04] P. Cimiano, A. Hotho, and S. Staab. Comparing Conceptual, Divise and Agglomerative Clustering for Learning Taxonomies from Text. In *Proceedings of the Sixteenth Eureopean Conference on Artificial Intelligence, ECAI'2004, including Prestigious Applicants of Intelligent Systems, PAIS 2004, Valencia, Spain, August 22-27, 2004*, pages 435–439. IOS Press, 2004.

[CMS99] R. Cooley, B. Mobasher, and J. Srivastava. Data Preparation for Mining World Wide Web Browsing Patterns. *Knowledge and Information Systems*, 1(1):5–32, 1999.

[Coo00] R. Cooley. *Web Usage Mining: Discovery and Application of Interesting Patterns from Web Data*. PhD thesis, University of Minnesota, May 2000.

[Coo03] R. Cooley. The Use of Web Structure and Content to Identify Subjectively Interesting Web Usage Patterns. *ACM Transactions on Internet Technology*, 3(2):93–116, 2003.

[CPY96] M. S. Chen, J. S. Park, and P. S. Yu. Data Mining for Path Traversal Patterns in a Web Environment. In *Proceedings of the Sixteenth International Conference on Distributed Computing Systems*, pages 385–392, 1996.

[CPY98] M. S. Chen, J. S. Park, and P. S. Yu. Efficient Data Mining for Path Traversal Patterns. *Knowledge and Data Engineering*, 10(2):209–221, 1998.

[CSM97] R. Cooley, J. Srivastava, and B. Mobasher. Web Mining: Information and Pattern Discovery on the World Wide Web. In *Proceedings of the Ninth IEEE International Conference on Tools with Artificial Intelligence (ICTAI'97)*, 1997.

[CSZG04] J. Chen, L. Sun, O. Zaïane, and R. Goebel. Visualizing and Discovering Web Navigational Patterns. In *Proceedings of the Seventh International Workshop on the Web and Databases (WebDB '04)*, pages 13–18. ACM Press, 2004.

[CT04] S. Chelcea and B. Trousse. Application of the 2-3 Agglomerative Hierarchical Classification on Web Usage Data. In D. Petcu, D. Zaharie, V. Negru, and T. Jebeleanu, editors, *Proceedings of SYNASC 2004, Sixth International Workshop on Symbolic and Numeric Algorithms for Scientific Computing*, pages 107–118. Editura Mirton, Timisoara, 26-30 September 2004.

[DG01] T. Draier and P. Gallinari. Characterizing Sequences of User Actions for Access Logs Analysis. In *Proceedings of the Eigth International Conference on User Modeling (UM 2001), Sonthofen, Germany, July 13-17*, volume 2109 of *LNCS*, pages 228–230. Springer, 2001.

[Did71] E. Diday. La méthode des nuées dynamiques. *Revue de Statistique Appliquée*, XIX(2):19–34, 1971.

[ESRR04] M. El-Sayed, C. Ruiz, and E. A. Rundensteiner. FS-Miner: Efficient and Incremental Mining of Frequent Sequence Patterns in Web Logs. In *Proceedings of the Sixth Annual ACM International Workshop on Web Information and Data Management (WIDM '04)*, pages 128–135. ACM Press, 2004.

[FL05] F. M. Facca and P. L. Lanzi. Mining Interesting Knowledge from Weblogs: a Survey. *Data & Knowledge Engineering*, 53(3):225–241, June 2005. In Press.

[FPSS96] U.M Fayad, G. Piatetsky-Shapiro, and P. Smyth. *From Data Mining to Knowldege Discovery: An Overview*, pages 1–34. In Advances in Knowledge Discovery and Data Mining. AAAI Press, Menlo Park, CA, 1996.

[FSS00] Y. Fu, K. Sandhu, and M. Shih. A Generalization-Based Approach to Clustering of Web Usage Sessions. In *Proceedings of the 1999 KDD Workshop on Web Mining, San Diego, CA*, volume 1836 of *LNAI*, pages 21–38. Springer, 2000.

[GCGR$^+$04] A. El Golli, B. Conan-Guez, F. Rossi, D. Tanasa, B. Trousse, and Y. Lechevallier. Les cartes topologiques auto-organisatrices pour l'analyse des fichiers logs. In *11èmes Rencontre de la Société Francophone de Classification*, Bordeaux, 8-10 September, 2004.

[GH98] D. Goldowitz and K. Hamre. The Cells and Molecules that Make a Cerebellum. *Trends in Neurosciences*, 21(9):375–382, 1998.

[Gia92] A. Giacometti. *Modèles Hybrides de l'Expertise.* PhD thesis, Telecom-Paris, 1992.

[Gol04] A. El Golli. *Extraction de données symboliques et cartes topologiques : Application aux données ayant une structure complexe.* PhD thesis, Université Paris IX Dauphine, June 2004.

[Goo05] Google search engine. `http://www.google.com/`, March 2005.

[Gor99] A. Gordon. *Classification.* Chapman and Hall, 1999.

[Gui04] F. Guillet. Mesure de la qualité des connaissances en ECD. Tutorial at EGC2004, Clermont Ferrand, 2004.

[Gus97] D. Gusfield. *Algorithms on Strings, Trees, and Sequences.* Cambridge University Press, 1997.

[HAC02] X. Huang, A. An, and N. Cercone. Comparison of Interestingness Functions for Learning Web Usage Patterns. In *Proceedings of the Eleventh International Conference on Information and Knowledge Management (CIKM '02)*, pages 617–620, New York, NY, USA, 2002. ACM Press.

[HBV04a] J. Huysmans, B. Baesens, and J. Vanthienen. The Influence of Caching on Web Usage Mining. In *Proceedings of Data Mining 2004: the Fifth International Conference on Data Mining, Text Mining and their Business Applications, Malaga, Spain.* Witpress, 15–17 September 2004.

[HBV04b] J. Huysmans, B. Baesens, and J. Vanthienen. Web Usage Mining: A Practical Study. In *Proceedings of the Twelfth Conference on Knowledge Acquisition and Management (KAM2004), Kule (Poland), May 13-15*, 2004.

[HK01] J. Han and M. Kamber. *Data Mining, Concepts and Techniques.* Morgan Kaufmann, 2001.

[HW05] E. Herder and H. Weinreich. Interactive web usage mining with the navigation visualizer. In *Proceedings of CHI '05 extended abstracts on Human factors in computing systems*, pages 1451–1454, New York, NY, USA, 2005. ACM Press.

[HWV03] B. Hay, G. Wets, and K. Vanhoof. Discovering Interesting Navigation on a Web Site Using Sequence Alignment Method Extended with an Interestingness Measure. In *Proceedings of the Intelligent Techniques for Web Personalization (ITWP '03)*, 2003.

[Jac98] M. Jaczynski. *Modèle et plate-forme à objets pour l'indexation des cas par situation comportementales: application à l'assistance à la navigation sur le Web.* PhD thesis, University of Nice Sophia Antipolis, December 1998.

[JK00] A. Joshi and R. Krishnapuram. On Mining Web Access Logs. In *ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, pages 63–69, 2000.

[JT98] M. Jaczynski and B. Trousse. WWW Assisted Browsing by Reusing Past Navigations of a Group of Users. In *Proceedings of the Advances in Case-Based Reasoning, Forth European Workshop, EWCBR-98, Dublin, Ireland, September 1998*, volume 1488 of *LNCS*, pages 160–171. Springer, 1998.

[KB00] R. Kosala and H. Blockeel. Web Mining Research: A Survey. *SIGKDD: SIGKDD Explorations: Newsletter of the Special Interest Group (SIG) on Knowledge Discovery & Data Mining, ACM*, 2(1):1–15, 2000.

[KF01] Y. Kagami and T. Furuichi. Investigation of Differentially Expressed Genes During the Development of Mouse Cerebellum. *Brain Research Gene Expression Patterns*, 1(1):39–59, August 2001.

[KNY00] H. Kato, T. Nakayama, and Y. Yamane. Navigation Analysis Tool Based on the Correlation Between Contets Distribution and Access Patterns. In *Worskhop on Web Mining for E-Commerce – Challenges and Opportunities, (KDD'00)*, pages 95–104, Boston, MA, USA, August 2000.

[Koh01] T. Kohonen. *Self-Organizing Maps.* Springer-Verlag, 2001.

[Kos] M. Koster. The Web Robots Database. `http://www.robotstxt.org/wc/active.html`.

[KR02] R. Kimball and M. Ross. *The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling.* Wiley, 2002.

[Kum04] H.-C. Kum. *Approximate Mining of Consensus Sequential Patterns.* PhD thesis, University of North Carolina, August 2004.

[LFGL99] R. Lipshutz, S. Fodor, T. Gingeras, and D. Lockhart. High density synthetic oligonucleotide arrays. *Nature Genetics*, 21(1):20–24, January 1999.

[LL03] C.T. Lewis and C. Lewis. Suffix Trees in Computational Biology. `http://homepage.usask.ca/~ctl271/857/suffix_tree.shtml`, 2003.

[LN99] B. Lavoie and H. F. Nielsen. Web Characterization Terminology & Definitions Sheet. `http://www.w3c.org/1999/05/WCA-terms/`, May 1999.

[LTTV03] Y. Lechevallier, D. Tanasa, B. Trousse, and R. Verde. Classification automatique : Applications au Web Mining. In Y. Dodge and G. Melfi, editors, *Méthodes et Perspectives en Classification*, pages 157–160. Presse Académiques Neuchâtel, 10-12 September 2003.

[LV04] Y. Lechevallier and R. Verde. Classiffication croisée d'un tableau de données symboliques : application à l'analyse du comportements des utilisateurs d'un site web. In *11èmes Rencontre de la Société Francophone de Classification*, Bordeaux, 8-10 September, 2004.

[LWP00] C-K. Lee, R. Weindruch, and T.A. Prolla. Gene-expression Profile of the Aging Brain in Mice. *Nature Genetics*, 25(3):294–297, 2000.

[Mal96] M. Malek. *Un modèle hybride de mémoire pour le raisonnement à partir de cas.* PhD thesis, University Joseph Fourrier Grenoble, October 1996.

[Mar04] R. Di Marco. *Tecniche di classificazione di dati simbolici in un contesto di Web Usage Mining.* PhD thesis, University "Federico II" Naples, November 2004.

[MBGMF04] D. Mladenic, J. Brank, M. Grobelnik, and N. Milic-Frayling. Feature Selection Using Linear Classifier Weights: Interaction with Classification Models. In *Proceedings of the Twenty-seventh Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Sheffield, UK, July 25-29, 2004*, pages 234–241. ACM, 2004.

[MBR04] C. Marquardt, K. Becker, and D. Ruiz. A Pre-Processing Tool for Web Usage Mining in the Distance Education Domain. In *Proceedings of the International Database Engineering and Applications Symposium (IDEAS'04)*, pages 78–87, 2004.

[MCS00] B. Mobasher, R. Cooley, and J. Srivastava. Automatic Personalization Based on Web Usage Mining. *Communications of the ACM*, 43(8):142–151, 2000.

[MDLN02] B. Mobasher, H. Dai, T. Luo, and M. Nakagawa. Discovery and Evaluation of Aggregate Usage Profiles for Web Personalization. *Data Mining and Knowledge Discovery*, 6(1):61–82, January 2002.

[MKS$^+$00] R. Matoba, K. Kato, S. Saito, C. Kurooka, C. Maruyama, Y. Sakakibara, and K. Matsubara. Gene Expression in Mouse Cerebellum During Its Development. *Gene*, 241(1):125–131, 2000.

[MPC99] F. Masseglia, P. Poncelet, and R. Cicchetti. An Efficient Algorithm for Web Usage Mining. *Networking and Information Systems Journal (NIS)*, 2(5–6):571–603, 1999.

[MS01] A. Maedche and S. Staab. Ontology Learning for the Semantic Web. *IEEE Intelligent Systems*, 16(2):72–79, 2001.

[MTP01a] F. Masseglia, M. Teisseire, and P. Poncelet. Real Time Web Usage Mining: a Heuristic based Distributed Miner (long). In *Proceedings of the Web Information Systems Ingineering Conference (WISE'01)*, Kyoto, Japan, December 2001.

[MTP01b] F. Masseglia, M. Teisseire, and P. Poncelet. Web Usage Mining Inter-Sites : Analyse du comportement des utilisateurs à impact immédiat. In *17^{ièmes} Journèes Bases de Données Avancées (BDA 2001)*, Agadir, Maroc, October 2001.

[MTT04a] F. Masseglia, D. Tanasa, and B. Trousse. Diviser pour découvrir. Une méthode d'analyse du comportement de tous les utilisateurs d'un site Web. *RSTI - Ingénierie des systèmes d'information (ISI)*, 9(1):61–83, 2004.

[MTT04b] F. Masseglia, D. Tanasa, and B. Trousse. Web Usage Mining: Sequential Pattern Extraction with a Very Low Support. In *Advanced Web Technologies and Applications: Proceedings of the Sixth Asia-Pacific Web Conference, APWeb 2004, Hangzhou, China, April 14-17*, volume 3007 of *LNCS*, pages 513–522. Springer-Verlag, 2004.

[NCS95] NCSA HTTPd LogOptions Directive. `http://hoohoo.ncsa.uiuc.edu/docs/setup/httpd/LogOptions.html`, 1995.

[Net05] Netcraft.com. March 2005 Web Server Survey. `http://news.netcraft.com/archives/2005/03/01/march_2005_web_server_survey_finds_60_million_sites.html`, March 2005.

[Nie00] Nielsen//NetRatings. Nielsen//Netratings Expands Global Internet Index to 17 Countries. `http://www.nielsen-netratings.com/pr/pr_001031.pdf`, October 31, 2000.

[NK02] O. Nasraoui and R. Krishnapuram. An Evolutionary Approach to Mining Robust Multi-Resolution Web Profiles and Context Sensitive URL Associations. *Journal of Computational Intelligence and Applications*, 2(3):339–348, 2002.

[OBHG03] D. Oberle, B. Berendt, A. Hotho, and J. Gonzalez. Conceptual User Tracking. In *Proceedings of Atlantic Web Intelligence Conference (AWIC'03)*, volume 2663 of *LNAI*, pages 155–164. Springer, 2003.

[Omn] Gene Expression Omnibus. National center for biotechnology information. `http://www.ncbi.nlm.nih.gov/geo/`.

[PE98] M. Perkowitz and O. Etzioni. Adaptive Web Sites: Automatically Synthesizing Web Pages. In *AAAI '98/IAAI '98: Proceedings of the Fifteenth National/Tenth International Conference on Artificial Intelligence/Innovative Applications of Artificial Intelligence*, pages 727–732. American Association for Artificial Intelligence, 1998.

[PHMAZ00] J. Pei, J. Han, B. Mortazavi-Asl, and H. Zhu. Mining Access Patterns Efficiently from Web Logs. In *Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 396–407, 2000.

[PK01] J. Punin and M. Krishnamoorthy. XGMML (eXtensible Graph Markup and Modeling Language). http://www.cs.rpi.edu/˜puninj/XGMML/draft-xgmml.html, June 2001.

[PPK$^+$00] G. Paliouras, C. Papatheodorou, V. Karkaletsis, P.Tzitziras, and C.D. Spyropoulos. Large-Scale Mining of Usage Data on Web Sites. In *Proceedings of the AAAI Spring Symposium on Adaptive User Interfaces*, pages 92–97, Stanford, California, 2000.

[RLG05] F. Rossi, Y. Lechevallier, and A. El Golli. Visualisation de la perception d'un site web par ses utilisateurs. *Revue des Nouvelles Technologies de l'Information (RNTI-E-3), numéro spécial EGC'2005*, 1:563–574, January 2005.

[SA96] R. Srikant and R. Agrawal. Mining Sequential Patterns: Generalizations and Performance Improvements. In *Proceedings of the Fifth International Conference on Extending Database Technology (EDBT'96)*, pages 3–17, Avignon, France, September 1996.

[SBP04] F. Silvestri, R. Baraglia, and P. Palmerini. On-line Generation of Suggestions for Web Users. *Journal of Digital Information*, 2(2):104–108, June 2004.

[SCDT00] J. Srivastava, R. Cooley, M. Deshpande, and P.-N. Tan. Web Usage Mining: Discovery and Applications of Usage Patterns from Web Data. *SIGKDD Explorations*, 1(2):12–23, 2000.

[SFW99] M. Spiliopoulou, L. C. Faulstich, and K. Winkler. A Data Miner Analyzing the Navigational Behaviour of Web Users. In *Proceedings of the Workshop on Machine Learning in User Modelling of the ACAI'99 International Conference*, Creta, Greece, July 1999.

[SK02] C. Shahabi and F. B. Kashani. A Framework for Efficient and Anonymous Web Usage Mining Based on Client-Side Tracking. In *WEBKDD 2001 - Mining Web Log Data Across All Customers Touch Points, Third International Workshop, San Francisco, CA, USA, August 26, 2001, Revised Papers*, volume 2356 of *LNCS*, pages 113–144. Springer, 2002.

[Spi99] M. Spiliopoulou. Data Mining for the Web. In J. Zytkow and J. Rauch, editors, *Principles of Data Mining and Knowledge Discover*, volume 1704 of *LNCS*, pages 588–589. Springer, 1999.

[SS73] Peter. Sneath and R. Sokal. *Numerical Taxonomy: The Principles and Practice of Numerical Classification.* W.H. Freeman, San Francisco, 1973.

[ST03] A. Santoni and D. Tanasa. Construction de la structure logique d'un site web. Technical report, Inria Sophia Antipolis, AxIS Project Team, 2003.

[Tan02] D. Tanasa. Lessons from a Web Usage Mining Intersites Experiment. In *Proceedings of the First International Workshop on Data Cleaning and Preprocessing of the ICDM02*, pages 99–107, Maebashi, Japon, 9 December 2002.

[TLT04] D. Tanasa, J. López, and B. Trousse. Extracting Sequential Patterns for Gene Regulatory Expressions Profiles. In *Knowledge Exploration in Life Science Informatics: International Symposium KELSI 2004, Milan, Italy, November 25-26, 2004*, volume 3303 of *LNCS*, pages 46–57. Springer-Verlag, 2004.

[TMGB05] B. Trousse, F. Masseglia, P. Gançarski, and O. Boussaid, editors. *Fouilles de données complexes*. RNTI. Cépaduès Editions, 2005. Proceedings of the Fouille de données complexes Workshop (EGC2005).

[TT01] D. Tanasa and B. Trousse. Web Access Pattern Discovery and Analysis based on Page Classification and on Indexing Sessions with a Generalised Suffix Tree. In *Proceedings of the Third International Workshop on Symbolic and Numeric Algorithms for Scientific Computing*, pages 62–72, Timisoara, Romania, October 2001.

[TT03] D. Tanasa and B. Trousse. Le prétraitement des fichiers logs web dans le "Web Usage Mining" multi-sites. In *Journées Francophones de la Toile (JFT'2003)*, pages 113–122, Tours, June - July 2003.

[TT04a] D. Tanasa and B. Trousse. Advanced Data Preprocessing for Intersites Web Usage Mining. *IEEE Intelligent Systems*, 19(2):59–65, March/April 2004.

[TT04b] D. Tanasa and B. Trousse. Data Preprocessing for WUM. *IEEE Potentials*, 23(3):22–25, August/September 2004.

[TTM04a] D. Tanasa, B. Trousse, and F. Masseglia. Classer pour Découvrir : une nouvelle méthode d'analyse du comportement de tous les utilisateurs d'un site Web. *Revue des Nouvelles Technologies de l'Information (RNTI), numéro spécial EGC'2004*, 2:549–560, January 2004.

[TTM04b] D. Tanasa, B. Trousse, and F. Masseglia. *Mesures de l'internet*, chapter Fouille de données appliquée aux logs web : état de l'art sur le Web Usage Mining, pages 126–143. Les Canadiens en Europe, 2004. Colloque "Mesures de l'internet", Nice, France, 12-14 Mai, 2003.

[Tur02] I. Turner. The One-Stop Portal. Line56, `http://www.line56.com/articles/default.asp?ArticleID=4075`, 8 October 2002.

[Ukk95] E. Ukkonen. On-line Construction of Suffix Trees. *Algorithmica*, 14:249–260, 1995.

[VBYA04] J. Velásquez, A. Bassi, H. Yasuda, and T. Aoki. Mining Web Data to Create Online Navigation Recommendations. In *Proceedings of the Fourth IEEE International Conference on Data Mining (ICDM'04), November 01 - 04, 2004, Brighton, United Kingdom*, pages 551–554. IEEE, 2004.

[W3C95] W3C. Logging Control In W3C httpd. `http://www.w3.org/Daemon/User/Config/Logging.html#common-logfile-format`, July 1995.

[WDM$^+$97] L. Wodicka, H. Dong, M. Mittmann, M.H. Ho, and D.J. Lockhart. Genome-wide Expression Monitoring in Saccharomyces Cerevisiae. *Nature Biotechnology*, 15(13):1359–1367, December 1997.

[Wei04] A. Weigend. People and Data: Understanding Customer Behavior. Invited Talk at the Fifth Data Mining Cup, Chemintz, Germany, June 2004.

[XD01] Y. Xiao and M. Dunham. Efficient Mining of Traversal Patterns. *Data and Knowledge Engineering*, 39(2):191–214, November 2001.

[ZRL96] T. Zhang, R. Ramakrishnan, and M. Livny. BIRCH: An Efficient Data Clustering Method for Very Large Databases. In H. V. Jagadish and I. S. Mumick, editors, *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data, Montreal, Quebec, Canada, June 4-6, 1996*, pages 103–114. ACM Press, 1996.

[ZSD03] O. Zaïane, S. Simoff, and C. Djeraba, editors. *Mining Multimedia and Complex Data, KDD Workshop MDM/KDD 2002, PAKDD Workshop KDMCD 2002, Revised Papers*, volume 2797 of *LNCS*. Springer, 2003.

[ZXH98] O. Zaïane, M. Xin, and J. Han. Discovering Web Access Patterns and Trends by Applying OLAP and Data Mining Technology on Web Logs. In *Proceedings of the Advances in Digital Libraries Conference*, pages 19–29, 1998.