Analyse du comportement des utilisateurs sur le Web

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

[1] LIM - [2] LIRMM - [3] IUT Aix-en-Provence - [4] PRiSM

[1] LIM - ESA CNRS 6077 - Université de la Méditerranée
Faculté des Sciences de Luminy, Case 901, 163 Avenue de Luminy,
13288 Marseille Cedex 9, France
E-mail: {poncelet,ciccheti}@lim.univ-mrs.fr

[2]LIRMM UMR CNRS 5506
161, Rue Ada
34392 Montpellier Cedex 5, France
E-mail: massegli@lirmm.fr

[4]Laboratoire PRiSM, Université de Versailles
45 Avenue des Etats-Unis
78035 Versailles Cedex, France

Résumé

Avec la popularité du World Wide Web (Web), de très grandes quantités de données comme l'adresse des utilisateurs ou les URL demandées sont automatiquement récupérées par les serveurs Web et stockées dans des fichiers access log. L'analyse de tels fichiers, appelée Web Usage Mining, offre des informations très utiles pour améliorer les performances du réseau, restructurer un site ou même cibler le comportement des clients dans le cadre du commerce électronique. Nous proposons dans cet article une approche et un système d'extraction de connaissances (WebTool) qui offre la possibilité d'extraire aussi bien des règles d'associations que des motifs séquentiels. En outre, pour illustrer une application des résultats obtenus par WebTool, nous montrons comment les motifs ou les règles intéressantes sont utilisés pour optimiser dynamiquement l'organisation hypertexte d'un serveur.

mots clés : data mining, web usage mining, motifs séquentiels, règles d'association.

Abstract

With the growing popularity of the World Wide Web (Web), large volumes of data such as user address or URL requested are gathered automatically by Web servers and collected in access log files. Analysis of server access data can provide significant and useful information for performance enhancement, restructuring a Web site for increased effectiveness, and customer targeting in electronic commerce. In this paper, we propose an integrated system (WebTool) for mining user patterns and association rules from one or more Web servers. Once the interesting patterns are discovered, we illustrate how they may be used to customize the server hypertext organization dynamically.

keywords: data mining, web usage mining, sequential patterns, association rules.

Catégorie : chercheur

1 Introduction

Motivé par des problèmes d'aide à la décision, le *data mining*, aussi appelé Extraction de Connaissances dans les Bases de Données (ECBD), a été largement étudié ces dernières années [11]. Dans ce domaine l'extraction de règles d'association a fait l'objet de nombreuses contributions [2, 3, 5, 11, 13, 27, 30, 32].

Initialement introduit dans [2], le problème de la recherche de règle d'association est souvent appelé problème du "supermarché" (*"market-basket" problem*) car les transactions opérées par les clients d'un magasin et dont la trace est stockée représentent une application typique pour le processus de découverte de connaissances. Dans un tel contexte, une règle d'association peut être par exemple: "85% des clients qui achètent les produits A et B achètent aussi C".
Dans [4], la problématique de la recherche de règles d'association est étendue pour détecter des comportements typiques dans le temps et le concept de motif séquentiel est introduit.
Avec la popularité du World Wide Web (Web), de très grandes quantités de données comme l'adresse des utilisateurs ou les URL demandées sont automatiquement récupérées par les serveurs Web et stockées dans des fichiers access log. L'analyse de tels fichiers peut offrir des informations très utiles pour, par exemple, améliorer les performances, restructurer un site ou même cibler le comportement des clients dans le cadre du commerce électronique.
La découverte d'information à partir du Web est généralement appelée Web Mining et peut recouvrir deux aspects: *Web content Mining* et *Web usage Mining* [9]. La première approche concerne la découverte et l'organisation d'informations extraites du Web. Par exemple, des approches basées sur les agents sont utilisées pour découvrir et organiser, de manière autonome, les informations extraites à partir du Web [17, 16, 24, 28] et des approches "bases de données" s'intéressent aux techniques d'intégration, d'organisation et d'interrogation de données hétérogènes et semi-structurées sur le Web [1, 22, 6, 12]. Le *Web usage Mining* s'intéresse par contre au problème de la recherche de motifs comportementaux des utilisateurs à partir d'un ou plusieurs serveur Web afin d'extraire des relations entre les données stockées. Bien que des outils d'analyse [15] existent pour indiquer, par exemple, le nombre d'accès à des URL ou la liste des URL les plus demandées, les rapports existant entre les ressources demandées et les accès des clients ne sont pas analysés par de tels outils dont les performances sont assez limitées [33].

Dans cet article, nous nous intéressons au problème de la découverte d'information dans un contexte de Web usage mining en attachant une attention particulière aux contraintes de temps. Nous proposons une approche générale et complète, permettant, à partir des données collectées, d'extraire aussi bien des règles d'association que des motifs séquentiels.

Dans notre contexte, en analysant les informations stockées sur le serveur Web, un exemple de règle d'association peut être: *50 % des clients qui ont visité les URL* `plaquette/info-f.html` *et* `labo/infos.html` *ont également visité* `situation.html` ou bien *85% des clients qui ont visité les URL* `iut/general.html departement/info.html` *et* `info/program.html` *ont également consulté l'URL* `info/debouches.html`.

L'extraction de motifs séquentiels permet, d'autre part, de mettre en évidence le type de relation suivant: *60 % des clients qui ont visité* `/jdk1.1.6/docs/api/Package-java.io.html` *et* `/jdk1.1.6/docs/api/java.io.BufferedWriter.html`*, ont également visité, dans les 30 jours suivants, l'URL* `/jdk1.1.6/docs/relnotes/deprecatedlist.html` ou *34 % des clients ont visité* `/relnotes/deprecatedlist.html` *entre le 20 septembre et le 30 novembre.*

Les règles et motifs découverts constituent une connaissance très utile pour optimiser dynamiquement l'organisation hypertexte d'un serveur. Très utiles dans des applications telles que la conception de catalogue "*on-line*" pour le commerce électronique, ces techniques exploitent les informations pertinentes sur le comportement des utilisateurs pour ré-organiser un site afin d'améliorer son efficacité. Par exemple, les informations proposées peuvent être utilisées par un serveur Proxy dans le but d'améliorer l'accès aux pages. Quand une requête atteint le serveur Web, la requête est envoyée au Proxy qui charge les documents désirés. Les motifs séquentiels découverts peuvent alors être utilisés pour pré-charger les document pour les utilisateurs [7].
L'approche proposée peut être mise en œuvre grâce à l'outil WebTool, qui permet le traitement des

données initiales, l'extraction et la gestion de connaissances et offre différentes possibilités pour en améliorer l'exploitation. Il permet en particulier, en reconnaissant un profil d'utilisateur typique, de restructurer dynamiquement un site.

L'article est organisé de la manière suivante. Le paragraphe 2 propose un exposé de la problématique concernant la recherche de règles d'association et de motifs séquentiels. Le problème est présenté dans son contexte initial, celui d'un supermarché, avec la description de l'algorithme générique utilisé par la plupart des approches. Dans le paragraphe 3, nous decrivons notre approche du Web Mining en détaillant les différentes étapes. Les expériences réalisées avec le prototype developpé et l'utilisation d'un mécanisme de mise à jour dynamique de pages Web sont présentées dans le paragraphe 4. Un bref état de l'art sur les approches de recherches d'informations dans le cadre du Web Mining est proposé au paragraphe 5. Enfin, dans le paragraphe 6, nous concluons en évoquant les suites de notre travail.

2 Recherche de règles d'association et de motifs séquentiels

Ce paragraphe expose et illustre la problématique liée à l'extraction de règles d'association et de motifs séquentiels dans le data mining. Il précise les concepts essentiels et propose un résumé des contributions apportées.

Dans [2], le problème de la recherche de règles d'association dans de grandes bases de données est défini de la manière suivante.

Définition 1 Soit $I = \{i_1, i_2, ..., i_m\}$, un ensemble de m achats (*items*). Soit $D = \{t_1, t_2, ...t_n\}$, un ensemble de n transactions ; chacune possède un unique identificateur appelé TID et porte sur un ensemble d'items (*itemset*) I. I est appelé un *k-itemset* où k représente le nombre d'éléments de I. Une transaction $t \in D$ contient un itemset I si et seulement si $I \subseteq t$. Le *support* d'un itemset I est le pourcentage de transaction dans D contenant I : $supp(I) = \|\{t \in D \mid I \subseteq t\}\|/\|\{t \in D\}\|$. Une règle d'association est une application conditionnelle entre les itemsets, $I_1 \Rightarrow I_2$ où les itemsets I_1, $I_2 \subset I$ et $I_1 \cap I_2 = \emptyset$. La *confiance* d'une règle d'association $r : I_1 \Rightarrow I_2$ est la probabilité conditionnelle qu'une transaction contienne I_2 étant donné qu'elle contient I_1. Le support d'une règle d'association est défini par $supp(r) = supp(I_1 \cup I_2)$.
Etant donné deux paramètres spécifiés par l'utilisateur, *minsupp* et *minconfiance*, le problème de la recherche de règles d'association dans une base de données D consiste à rechercher l'ensemble des itemsets fréquents dans D, i.e. tous les itemsets dont le support est supérieur ou égal à *minsupp*. Puis, à partir de cet ensemble, générer toutes les règles d'association dont la confiance est supérieure à *minconfiance*.

Pour étendre la problématique précédente à la prise en compte du temps des transactions, les mêmes auteurs ont proposé dans [4] la notion de séquence définie de la manière suivante :

Définition 2 Une *transaction* constitue, pour un client C, l'ensemble des items achetés par C à une même date. Dans une base de données client, une transaction s'écrit sous forme d'un triplet : <id-client, id-date, itemset>. Un *itemset* est un ensemble non vide d'items noté $(i_1 i_2 \dots i_k)$ où i_j est un *item* (il s'agit de la représentation d'une transaction non datée). Une *séquence* est une liste ordonnée, non vide, d'itemsets notée $< s_1 s_2 \dots s_n >$ où s_j est un itemset (une séquence est donc une suite de transactions avec une relation d'ordre entre les transactions). Une *séquence de données* est une séquence représentant les achats d'un client. Soit T_1, $T_2, \dots$, T_n les transactions d'un client, ordonnées par date d'achat croissante et soit itemset(T_i) l'ensemble des items correspondants à T_i, alors la séquence de données de ce client est $< itemset(T_1)\ itemset(T_2) \dots\ itemset(T_n) >$.

Exemple 1 *soit C un client et S=< (3) (4 5) (8) >, la séquence de données représentant les achats de ce client. S peut être interprétée par "C a acheté l'item 3, puis en même temps les items 4 et 5 et enfin l'item 8".*

La notion de séquence telle qu'elle a été introduite présente cependant quelques limites pour certains types d'application. Aussi la notion de séquences généraliséees a-t-elle été introduite dans [31] pour pallier les restrictions suivantes :

- **Absence de contraintes de temps.** Il peut être nécessaire d'exprimer des contraintes temporelles entre transactions car, dans certains cas, une transaction survenue trop tôt ou trop tard perd toute sa pertinence.
- **Rigidité de la définition des transactions.** Dans de nombreuses applications, si l'intervalle de temps entre 2 transactions est de courte durée, elles doivent être assimilées à une seule et même transaction.

De manière plus formelle, le problème de la recherche de séquences généralisées est défini dans [31] de la façon suivante :

Définition 3 Soient *minGap* une durée minimum, *maxGap* une durée maximum et *windowSize* une durée de rectification (également notée ws), spécifiées par l'utilisateur. Une séquence de données $d =< d_1...d_m >$ supporte une séquence $s =< s_1...s_n >$ s'il existe des entiers $l_1 \leq u_1 < l_2 \leq u_2 < ... < l_n \leq u_n$ tels que :

1. $s_i \subset \cup_{k=l_i}^{u_i} d_k$, $1 \leq i \leq n$;
2. date(d_{u_i}) - date(d_{l_i}) $\leq$ *windowSize*, $1 \leq i \leq n$;
3. date(d_{l_i}) - date($d_{u_{i-1}}$) $>$ *min-gap*, $2 \leq i \leq n$;
4. date(d_{u_i}) - date($d_{l_{i-1}}$) $\leq$ *max-gap*, $2 \leq i \leq n$.

Le *support* de s, noté *supp*(s), est le pourcentage de toutes les séquences dans D qui supportent s. Si *supp*(s) $\geq$ *minsupp*, avec une valeur de support minimum *minsupp*, la séquence s est dite *fréquente*.

Ce concept de séquences généralisées permet une manipulation plus souple des transactions des clients, dans la mesure où il est désormais possible de :

- regrouper des achats lorsque leurs dates sont assez proches via la contrainte de WindowSize,
- considérer des itemsets (achats) comme trop rapprochés pour apparaître dans la même séquence fréquente avec la contrainte de minGap,
- considérer des itemsets (achats) comme trop éloignés pour apparaître dans la même séquence fréquente avec la contrainte de maxGap.

Client	Date	Items
C_1	1	Ringworld
C_1	2	Foundation
C_1	15	Ringworld Engineers, Second foundation
C_2	1	Foundation, Ringworld
C_2	20	Foundation and Empire
C_2	50	Ringworld Engineers

FIG. 1 – *Base de données exemple*

Exemple 2 Considérons l'exemple suivant, extrait de [31], qui illustre la prise en compte des contraintes de temps sur une base de données réduite à 2 clients (C.f. Figure 2). Avec un support minimum strictement supérieur à 50%, i.e. pour qu'une séquence soit supportée elle doit être vérifiée par les deux

séquences de la base de données les motifs séquentiels sont d'après la définition 3: <(Ringworld) (Ringworld Engineers)> et <(Foundation) (Ringworld Engineers)>.
Si nous considérons maintenant une durée de 7 jours pour windowSize, la séquence fréquente suivante apparaît: <(Foundation, Ringworld) (Ringworld Engineers)> car les deux premières transactions du client C_1, survenues à un jour d'écart, peuvent être appréhendées comme une seule transaction. La séquence précédente est alors vérifiée pour les deux clients et son support est de 100%. L'ensemble des motifs séquentiels est alors le suivant : E=<(Foundation, Ringworld) (Ringworld Engineers)>, car nous avons <(Foundation) (Ringworld Engineers)>$\prec$<(Foundation, Ringworld) (Ringworld Engineers)> et <(Ringworld) (Ringworld Engineers)>$\prec$<(Foundation, Ringworld) (Ringworld Engineers)>.
Par contre, avec une contrainte de maxGap fixée à 30 jours, les séquences trouvées précédemment ne sont plus fréquentes car elles ne sont plus vérifiées par C_2 (49 jours s'écoulant entre les deux transactions de la séquence).

Un algorithme générique

La technique généralement utilisée par les algorithmes de recherche de séquences ou de règles d'association est basée sur une création de candidats, suivie du test de ces candidats pour confirmer leur fréquence dans la base. Si le coût de opérations de Lecture/Ecriture sur la base n'était pas si élevé, une technique rapide pour déterminer les fréquents consisterait à prendre la séquence réduite à un item et essayer de l'étendre le plus possible et de recommencer pour chaque item. Cela impliquerait un trop grand nombre de passes sur la base de données et donc des temps de réponses catastrophiques.
Pour permettre de garder un nombre de passes acceptable sur la base de données, la méthode *Générer-Elaguer* propose de faire croître la taille des fréquents découverts dans la base de données à chaque étape de son algorithme. Cette méthode construit un sur-ensemble C_i des fréquents qui seront trouvés à l'étape i (la construction de C_i respecte des propriétés qui assurent que tous les i-fréquents appartiennent à C_i). C_i est désigné comme l'ensemble des candidats. Cet ensemble est ensuite élagué par vérification dans la base de données, afin de ne garder que les séquences fréquentes. Après cette étape l'ensemble C_i devient L_i (l'ensemble des fréquents), L_i est utilisé pour construire le sur-ensemble C_{i+1}des fréquents de l'étape $i+1$ et l'algorithme recommence une nouvelle étape. La méthode prend fin quand L_i est vide, c'est à dire quand tous les candidats ont été invalidés. Cette méthode est mise en œuvre par l'algorithme générique: ALGOGENERIQUE (algorithme 1).

function *algoGenerique*
Input : Un support minimum *minsupp* et une base de données D.
Output : L'ensemble L des séquences ayant une fréquence d'apparitions supérieure à *minsupp*.
$k = 1$;
// Les itemsets fréquents
$C_1 = \{\{< i >\}/i \in I\}$;
foreach $d \in D$ **do** $compterSupport(C_1, minsupp, d)$;
$L_1 = \{c \in C_1/Support(c) > minsupp\}$;
while $L \neq \emptyset$ **do**
  $genererCandidats(C_k)$;
  foreach $d \in D$ **do** $compterSupport(C_k, minsupp, d)$;
  $L_k = \{c \in C_k/Support(c) > minsupp\}$;
  $k = k + 1$;
endwhile
return $L = \bigcup_{j=0}^{k} L_j$;
end function *algoGenerique*

Algorithme 1: *Algorithme générique*

Cet algorithme utilise deux fonctions indispensables:

- compterSupport. Cette fonction est destinée à incrémenter le support des candidats contenus dans C_k à partir de la séquence de données d et en fonction du support minimum *minsupp*.
- genererCandidats. Cette fonction a pour but de créer tous les k-candidats susceptibles d'être fréquents (donc tous les k-candidats susceptibles de devenir des k-fréquents) à partir d'un ensemble de (k-1)-fréquents.

3 Une approche de Web mining

La démarche que nous proposons vise aussi bien l'extraction de règles d'associations que de motifs séquentiels dans le cadre du Web mining. Ces principes généraux, illustrés par la figure 2, sont similaires à ceux du processus d'extraction de connaissances exposés dans [11]. La démarche se décompose en trois phases principales. Tout d'abord, à partir d'un fichier de données brutes, un pré-traitement est nécessaire pour éliminer les informations inutiles. Dans la deuxième phase, à partir des données transformées, des algorithmes de data mining sont utilisés pour extraire les itemsets ou les séquences fréquents. Enfin, l'exploitation par l'utilisateur des résultats obtenus est facilitée par un outil de requête et de visualisation.

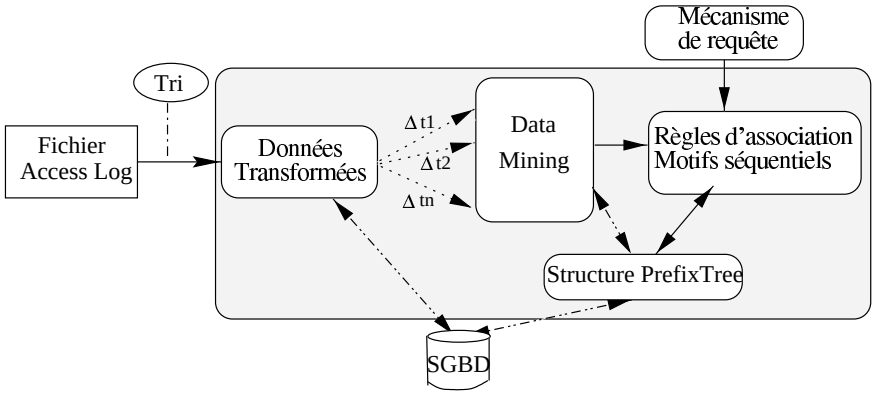

FIG. 2 – *Principe généraux de la démarche*

Les différentes phases introduites sont largement détaillées dans les paragraphes suivants.

3.1 Pré-traitement des données

Dans notre contexte de Web mining, les données brutes sont collectées dans des fichiers access log des serveurs Web. Une entrée dans le fichier access log est automatiquement ajoutée chaque fois qu'une requête pour une ressource atteint le serveur Web (*demon http*). Spécifiée par le CERN et la NCSA [8], une entrée contient des enregistrents formés de 7 champs séparés par des espaces [26]:

```
host user authuser [date:time] ''request'' status bytes
```

Le tableau de la figure 3 décrit chaque champ et son contenu[1].
La figure 3.1 illustre un extrait du fichier access log du serveur Web de l'IUT d'Aix en Provence.

Deux types de traitements sont effectués sur les entrées du serveur log. Tout d'abord, le fichier access log est trié par adresse et par transaction. Ensuite une étape d'élimination des données "non-intéressantes"

1. Dans la suite de cette article, nous considérons qu'une entrée dans le fichier access log est réduite à l'adresse IP d'où provient la requête, à l'URL obtenue et au temps d'accès.

Champ	Contenu
`host`	Le nom ou l'adresse IP du client.
`user`	Toute information retournée par *identd* pour cet utilisateur, ou "-" par défaut.
`authuser`	l'identificateur utilisateur si celui-ci l'a envoyé. sinon "-".
`date`	La date dans le format JJ/Mois/Année.
`time`	L'heure dans le format hh:mm:ss.
`request`	La première ligne de la requête HTTP faite par le client (par exemple, PUT ou GET suivi par le nom de l'URL demandé).
`status`	Le code renvoyé par le serveur en réponse à cette requête, ou "-" par défaut.
`bytes`	Le nombre total d'octets envoyés (sans compter l'entête HTTP), ou "-" par défaut.

FIG. 3 – *Description des champs d'une entrée*

```
132.208.12.150 - - [29/Nov/1998:18:02:26 +0200] "GET /info/COINETUD.gif HTTP/1.0 " 200 1159
132.208.12.150 - - [29/Nov/1998:18:02:27 +0200] "GET /info/CONTACTER.gif HTTP/1. 0" 200 1137
132.208.12.150 - - [29/Nov/1998:18:02:28 +0200] "GET /info/DEBOUCHES.gif HTTP/1. 0" 200 1150
132.208.12.150 - - [29/Nov/1998:18:02:30 +0200] "GET /info/RECRUT.gif HTTP/1.0" 200 1141
132.208.12.150 - - [29/Oct/1998:18:03:07 +0200] "GET /info/recrut.html HTTP/1.0" 200 1051
132.208.127.200 - - [16/Oct/1998:20:34:32 +0100] "GET /geaaix/home.html HTTP/1.0 " 200 14617
148.241.148.34 - - [31/Oct/1998:01:17:40 +0200] "GET /info/index.html HTTP/1.0" 304 -
148.241.148.34 - - [31/Oct/1998:01:17:42 +0200] "GET /info/recrut.html HTTP/1.0" 304 -
192.70.76.73 - - [22/Nov/1998:11:06:11 +0200] "GET /info/program.html HTTP/1.0" 200 4280
192.70.76.73 - - [22/Nov/1998:11:06:12 +0200] "GET /info/MATIERES.gif HTTP/1.0" 200 2002
192.93.19.14 - - [07/Dec/1998:11:44:15 +0200] "GET /queldept.html HTTP/1.0" 200 5003
```

FIG. 4 – *Exemple de fichier access log*

est réalisée.
La plupart des outils d'analyse du Web profitent de cette étape pour éliminer des requêtes concernant des pages possédant des graphiques, des vidéos, des scripts CGI ou du son (par exemple, en éliminant les fichiers suffixés par *.GIF, .JPEG*) ou bien les requêtes issus d'agents ou de testeurs de liens. Même si WebTool offre également cette possibilité, nous préférons conserver ces informations comme dans le système WebLogMiner [33]. En effet, ces données apportent des renseignements intéressants concernant aussi bien la structure du site Web, la motivation d'un utilisateur ou les performances du trafic. Par exemple, le fait de conserver des requêtes générées par des agents est utile pour analyser le comportement de l'agent et comparer le trafic généré par ces agents avec le reste du trafic [33]. De la même manière, si un utilisateur pose des requêtes pour des pages graphiques, certaines de ces requêtes sont importantes pour appréhender les actions de l'utilisateur.

Bien entendu, un tel choix nécessite la mise en œuvre d'algorithmes d'extraction efficaces car la taille du fichier access log reste très grande (au cours de nos expériences, nous avons constaté que la suppression des URL concernant des images réduisait la taille du fichier log de 40% à 85%). En outre, il est nécessaire d'offrir à l'utilisateur un outil de visualisation pour éliminer les données qui ne l'intéressent pas lors d'une analyse particulière.

Au cours de la phase de tri et afin de rendre plus efficace le traitement de l'extraction de données (C.f. paragraphe 3.2.3), les URL et les clients sont codés sous forme d'entiers. Toutes les dates sont également traduites en temps relatif par rapport à la plus petite date du fichier.

L'étape de pré-traitement des données offre également la possibilité de regrouper les entrées qui sont suffisamment proches. Contrairement au problème du supermarché où chaque transaction est définie comme un ensemble d'achats effectués par un client, les entrées dans le fichier log sont toutes des transactions séparées. De la même manière que [9], nous proposons de regrouper les entrées qui sont suffisamment proches en utilisant une contrainte de temps (Δt) spécifiée par l'utilisateur. Ainsi, une entrée dans le fichier log est définie de la manière suivante :

Définition 4 Soit *Log* un ensemble d'entrées dans le fichier access log. Une entrée g, $g \in Log$, est un tuple $g =< ip_g, \{(l_1^g.URL, l_1^g.time), ..., (l_m^g.URL, l_m^g.time)\} >$ tel que pour $1 \leq k \leq m$, $l_k^t \in Log$, $l_k^g.ip = ip_g$, $l_k^g.URL$ doit être unique et $l_{k+1}^g.time - l_k^g.time \leq \Delta t$.

A l'heure actuelle, chaque valeur de Δt engendre un nouveau fichier résultat contenant les transactions codées et regroupées en fonction de la fenêtre de temps. La figure 5 illustre un exemple de fichier obtenu après la phase de pré-traitement.

Client	Date	URL traduite
1	01	10,30,40
1	02	20,30
2	11	10
2	12	30,60
2	23	20,50
3	01	10,70
3	12	30
3	15	20,30

FIG. 5 – *Exemple de fichier résultat issu de la phase de pré-traitement*

3.2 Outils d'Extraction

A partir des données transformées issues de la première étape, deux techniques d'extraction de connaissance peuvent être appliquées selon les besoins de l'analyste.

3.2.1 Extraction de motifs séquentiels

La prise en compte du temps pour la recherche de motifs nécessite de redéfinir la notion de séquence dans le cadre du Web Mining.

Définition 5 Soit TT l'ensemble de toutes les transactions temporelles. Une transaction temporelle t, $t \in TT$, est un triplet $t =< ip_t, UT_t, time_t >$ où UT_t, l'ensemble *URL-Time* pour t, est défini par $UT_t = \{(l_1^t.URL, l_1^t.time), ..., (l_m^t.URL, l_m^t.time)\}$, tels que pour $1 \leq k \leq m$, $l_k^t \in Log$, $l_k^t.ip = ip_t$, $l_k^t.URL$ doit être unique in UT_t, $l_{k+1}^t.time - l_k^t.time \leq \Delta t$, et $time_t = max_{1 \leq i \leq m} l_i^t.time$.

Les UT-séquences sont construites à partir d'itemsets où chaque item est une URL obtenue par le client dans une transaction.

Définition 6 Soit l'ensemble de transactions $T' = \{t_i \in T | 1 \leq i \leq k\}$, une *UT-squence* S pour T' est définie par : $S =< UT_{t_1}, ..., UT_{t_k} >$, où $time_{t_i} < time_{t_{i+1}}$, pour $1 \leq i \leq k-1$. Une *k-UT-séquence*, ou *k-séquence*, est une séquence de k URL.
Une UT-séquence, S_c, pour un client c est appelée une *séquence de données* et est définie par : $S_c =< UT_{t_1}, UT_{t_2}, ..., UT_{t_n} >$ où, pour $1 \leq i \leq n$, $t_i \in T_c$, et T_c représente l'ensemble de toutes les transactions temporelles du client c, i.e. $T_c = \{t \in T | ip_t = ip_c\}$.

Pour déterminer si une séquence de données est fréquente dans la base de données D, les différents paramètres temporels ws, *min-gap*, et *max-gap* ainsi que le support sont spécifiés par l'utilisateur. Une séquence est alors reconnue fréquente si elle respecte la définition 3.

Exemple 3 Considérons la base de la figure 5, avec les paramètres utilisateurs spécifiés de la manière suivante : $minsupp = 50\%$ indiquant qu'il faut qu'une séquence soit vérifiée par au moins deux clients de la base pour être fréquente, $ws = \emptyset$, $min\text{-}gap = \emptyset$ et $max\text{-}gap = \infty$, les quatre séquences suivantes sont obtenues : $\{< (10)\ (30) >, < (10)\ (20) >, < (20\ 30) >, < (30)\ (20) >\}$.

La figure 6 illustre un exemple d'utilisation de WebTool pour extraire des motifs séquentiels.

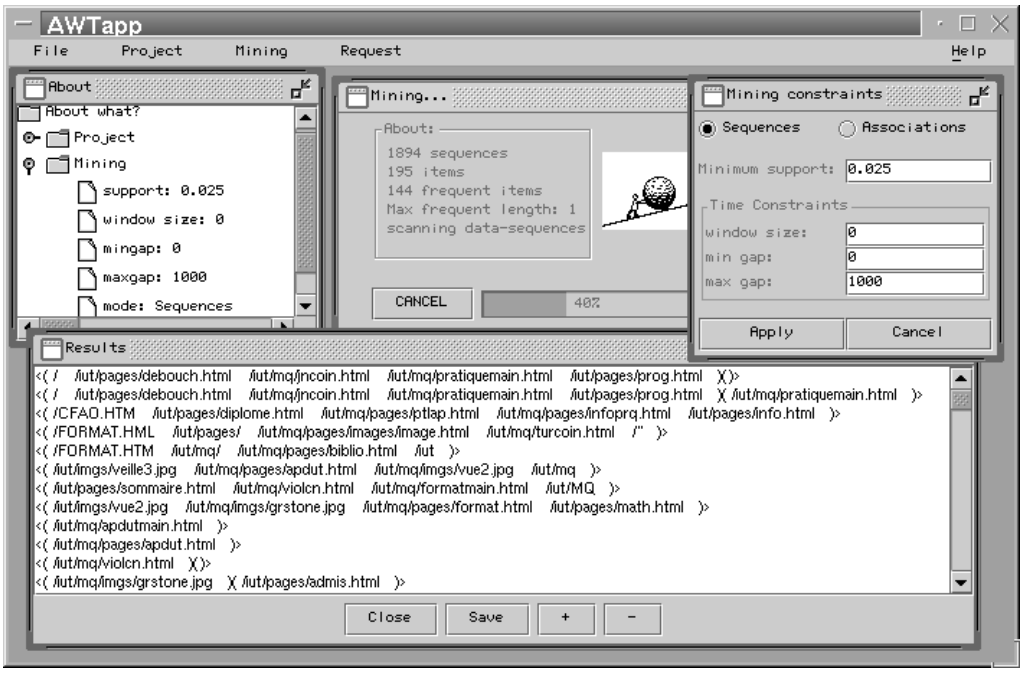

FIG. 6 – *WebTool lors de l'extraction de motifs séquentiels*

3.2.2 Extraction de règles d'association

Les techniques de découverte de règles d'association sont généralement appliqués dans des bases de données où chaque transaction est composée d'un ensemble d'items. Comme nous l'avons vu dans la phase de pré-traitement, il est nécessaire dans le cadre du Web Mining de regrouper les items entre-eux. Ainsi, la définition suivante précise les transactions manipulées lors de la recherche de règles d'association :

Définition 7 Soit TA l'ensemble de toutes les transactions d'association. Une transaction d'association t, $t \in TT$, est un tuple $t =< ip_t, UT_t >$ où UT_t, l'ensemble $URL\text{-}Time$ pour t, est défini par $UT_t = \{(l_1^t.URL, l_1^t.time), ..., (l_m^t.URL, l_m^t.time)\}$, tels que pour $1 \leq k \leq m$, $l_k^t \in Log$, $l_k^t.ip = ip_t$, $l_k^t.URL$ doit être unique dans UT_t, $l_{k+1}^t.time - l_k^t.time \leq \Delta t$ et $l_{k-1}^t.URL <_{prec} l_k^t.time$.

En d'autres termes, une transaction d'association ne tient pas compte du temps entre les transactions et pour chaque transaction les URL manipulées sont triées par ordre croissant.

Client	URL traduite
1	10,20,30,40
2	10,20,30,50,60
3	10,20,30,70

FIG. 7 – *Exemple de transactions d'association*

Exemple 4 A partir de la base exemple illustrée par la figure 5, une nouvelle base de transactions d'associations est obtenue. Elle est illustrée par la figure 7. Sur cette nouvelle base, l'extraction de règles d'association avec un support minimal de 50% rend l'itemset fréquent (10 20 30).

3.2.3 Utilisation d'un algorithme et d'une structure efficace pour l'extraction

Etant donné qu'une grande quantité d'informations est récoltée par les serveurs Web et stockée dans les fichiers access log, des algorithmes efficaces pour extraire des motifs séquentiels ou des règles d'association sont nécessaires.

L'algorithme PSP (*Prefix-tree for Sequential Patterns*) que nous utilisons dans le système WebTool a tout d'abord été défini pour la recherche de motifs séquentiels [21].
Le principe retenu par PSP est celui de la méthode de *Générer-Elaguer* décrite dans [31] : à chaque passe, un ensemble de candidats est généré et testé sur la base. Pour stocker et rechercher efficacement les candidats dans la base, nous utilisons une structure PrefixTree. Extension de la structure proposée par [25], cette structure offre deux avantages principaux par rapport à la structure de *hash-tree* utilisée dans GSP [31] ou Apriori [3] :

1. les itemsets ou séquences fréquentes sont stockées dans le même arbre afin d'optimiser la génération des candidats ;
2. contrairement à GSP qui nécessite une phase supplémentaire pour gérer les contraintes de temps, celles-ci peuvent être prises en compte directement lors du parcours de l'arbre.

L'efficacité d'une structure PrefixTree pour la recherche de règles d'association a été montré dans [25] et nous avons montré, dans [21, 20], qu'une extension de cette structure pour prendre en compte la notion du temps était également très efficace. La figure 8 illustre les temps de réponse de PSP, en faisant varier le support minimum, pour obtenir les séquences fréquentes sur un fichier access log de 85 MØ.
Etant donné que le problème de la recherche de règles d'association est une réduction du problème de la recherche de motifs séquentiels, le principe retenue par WebTool est d'utiliser la même structure et les mêmes algorithmes pour obtenir des règles d'association. L'adaptation de PSP est prise en compte en considérant que toutes les transactions ont eu lieu en même temps. Ainsi, lors de l'application de PSP, les temps des différentes transactions ne sont plus considérés et les résultats obtenus ne sont plus des séquences fréquentes mais des itemsets fréquents. La génération des règles basées sur ces itemsets est réalisée par l'outil de visualisation.

3.3 Outils de Visualisation

Devant le très grand nombre de motifs ou d'itemsets obtenus, il apparaît indispensable de proposer un mécanisme de requête pour offrir à l'utilisateur la possibilité de mieux analyser les informations découvertes lors la phase précédente. Un tel mécanisme peut intervenir de deux manières différentes :

1. lors de la phase d'obtention des règles d'association, i.e. après application des algorithmes de recherche de motifs ou d'itemsets fréquents. Il s'agit donc d'extraire des règles à partir des connaissances acquises par les outils d'extraction.

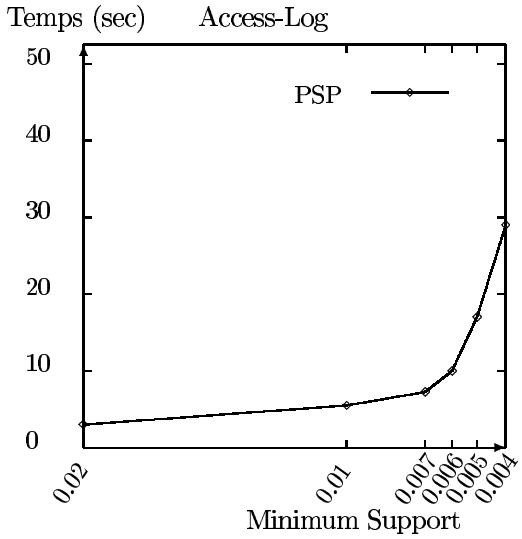

FIG. 8 – *Temps d'exécution*

2. lors de l'extraction pour optimiser les temps de traitement réduisant la portion de base à extraire. Ce principe rejoint, par exemple, la phase de pré-traitement où les URL concernant des pages graphiques peuvent être éliminés de la base.

Dans le système WebTool nous privilégions, pour le moment, la première approche dans la mesure où nous considérons que l'utilisateur peut vouloir affiner ces recherches sans recommencer complètement le processus d'extraction.

A l'heure actuelle, de nombreux outils d'extraction proposent d'interroger les résultats via un langage de requête graphique [14]. Le système WebTool s'inspire de ces travaux et propose à l'utilisateur la possibilité de spécifier le type de règles désiré de la même manière que les "*templates*" utilisés dans le système Tasa [19, 18]. Tout à fait comparable à un langage SQL-like, l'utilisateur peut préciser :

- les antécédents de la règle,
- les conséquents de la règle,
- une restriction sur les dates ou le domaine des adresses IP (par exemple, l'utilisateur ne peut être interessé que par les accès au serveur des clients du domaine "**.fr**").

La requête suivante illustre une traduction dans un langage SQL-like d'une requête demandant les règles obtenues à partir des motifs séquentiels où la partie antécédent contient au moins les URL `/FORMAT.HTM` et `/FORMAT.HML`. La figure 9 illustre le même exemple mais formulée via l'interface utilisateur de notre système.

```
SELECT sequential-patterns-rules (
        ANTECEDENT = * /FORMAT.HTM /FORMAT.HML *
        CONSEQUENT = /CFAO.HTM * /iut/pages/
       )
FROM access log
WHERE support = 1.0 AND confiance = 5.5
      AND domain = ''fr'' AND date > 11/12/1998
      AND window-size = 10 AND min-gap = 2
```

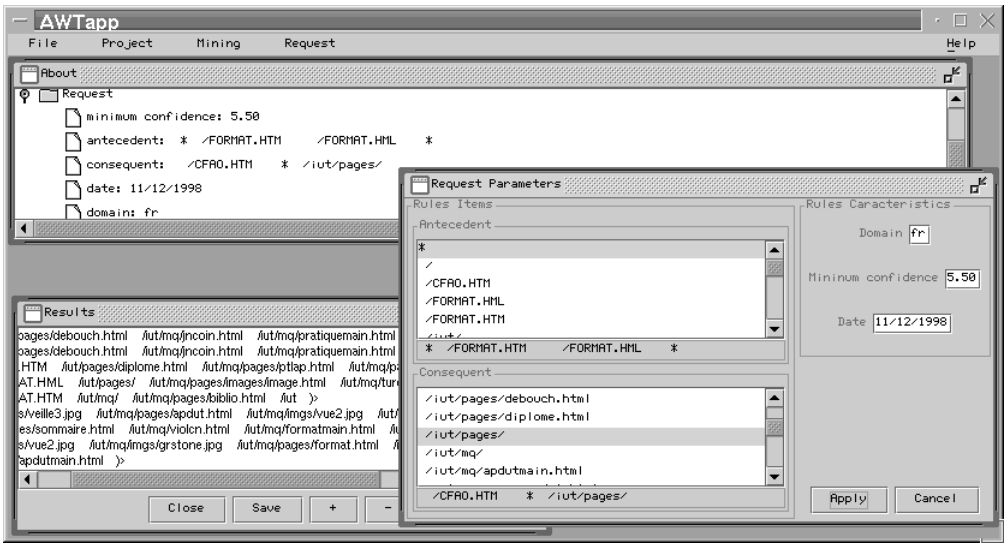

FIG. 9 – *Exemple de requête dans WebTool*

4 Expérimentations

Le prototype du système WebTool est développé sur une station Ultra Sparc. Les algorithmes de recherche de séquences et de règles d'association ont été développés en utilisant GNU C++ avec la bibliothèque STL. l'interface utilisateur, l'outil de visualisation et la phase de prétraitement ont été réalisés en Java (JDK1.1.6 et Swing-1.1). Nous avons utilisé le système WebTool pour analyser des fichiers access log du Laboratoire LIRMM et de l'IUT d'Aix en Provence.

Exemple 5 Considérons la règle d'association suivante obtenue par le processus d'extraction sur le fichier access log du LIRMM :

`<(lirmm/plaquette/info-f.html lirmm-infos.html)>`
$\Rightarrow$ `<(situ.html /autour.html mtp/index.html)>` *conf = 13*

Elle indique que 13% des usagers qui ont recherché des informations sur le laboratoire LIRMM et plus particulièrement sur l'informatique, ont cherché à connaître la situation géographique du laboratoire (`situ.html`), comment accéder au laboratoire (`autour.html`) et des renseignements sur la ville de Montpellier (`mtp/index.html`).

En spécifiant les paramètres *min-gap*=2 jours, *max-gap*=5 jours, *ws*=1 jours, *minsupp*=2% et *min-confiance*=60%, nous avons obtenu, à partir des motifs séquentiels, la règle suivante :

`<(/jdk1.1.6 /jdk1.1.6/docs/index.html), (/jdk1.1.6/docs/api/packages.html)>`
$\Rightarrow$ `<(/jdk1.1.6/docs/relnotes/deprecatedlist.html)>` *conf = 67, supp = 2,3*

qui indique que 67% des gens qui ont consulté les URL (`/jdk1.1.6, /jdk1.1.6/docs/index.html`) la même journée, l'URL (`/jdk1.1.6/docs/api/packages.html`) plus de 2 jours après, ont également

consulté l'URL (`/jdk1.1.6/docs/relnotes/deprecatedlist.html`) dans les 5 jours et qu'ils sont 2,3% à avoir consulté les quatre URL.

Prise en compte des règles obtenues

Conjointement au système WebTool, nous avons développé un générateur de création de liens dynamiques dans des pages Web à partir des règles obtenues lors du processus d'extraction. Le but du générateur est de reconnaître un utilisateur et de vérifier son parcours dans les pages d'un serveur. Lorsque celui-ci correspond à une règle d'association ou à un motif séquentiel déterminé par WebTool, les pointeurs des pages sont dynamiquement mis à jour.

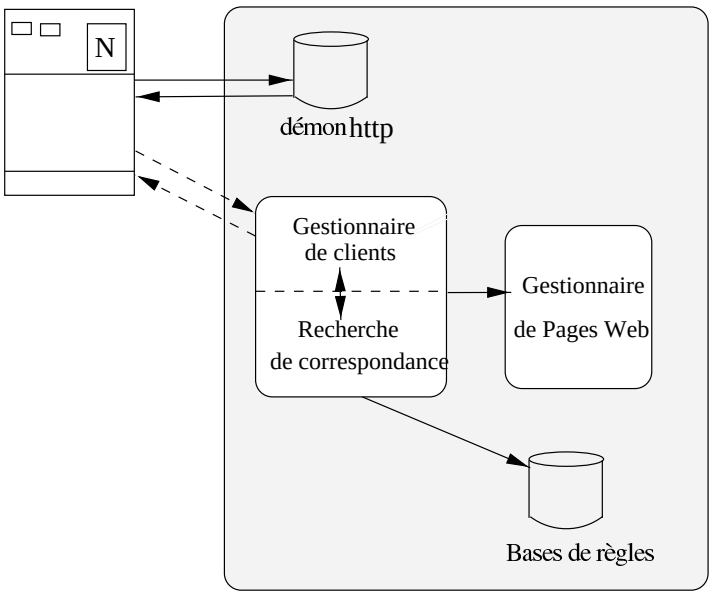

FIG. 10 – *Architecture générale*

L'architecture fonctionnelle de l'approche est décrite dans la figure 10. Le serveur Web (démon *http*) réagit à l'envoi d'une requête par un client en lui retournant une page contenant une applet (C.f. Figure 11). Celle-ci est chargée de se connecter au serveur de clients (*gestionnaire de clients*) pour lui transmettre l'adresse Internet du client et la page demandée. A l'heure actuelle, le gestionnaire de clients est une application java qui fonctionne sur la même machine que le serveur *http* pour permettre à l'applet l'utilisation d'un mécanisme client/serveur.
Lors de la réception de l'adresse et de la page, le gestionnaire de clients examine le comportement du client via le module de recherche de correspondances qui est chargé de vérifier si le comportement du client est en adéquation avec une règle préalablement extraite par le processus de data mining.
Pour déterminer si une entrée est reconnue par une règle, elle doit vérifier la définition suivante :

Définition 8 Une entrée pour un client C_i est un couple défini par $E_{C_i} =< id_{C_i}, \{u_1, u_2, ...u_n\} >$

```
<HTML>
<APPLET code="AppletClient.class">
<PARAM  name="file" value="info/genera.html" /PARAM>
</APPLET>
</HTML>
```

FIG. 11 – *Exemple de page renvoyée au client par le démon http*

tel que pour $1 \leq k \leq n$, u_k est une URL obtenue par le client. Une règle R est un triplet défini par $R =< (a_1, a_2, ...a_i), (c_1, c_2, ..., c_j), conf_R >$ tel que pour $1 \leq k \leq i$, a_k représente les antécédents de la règles, i.e. des URL, $1 \leq k \leq j$, c_k représente les conséquents de la règle et $conf$ est la *confiance*. Soit $conf$, la valeur minimum pour qu'une règle s'applique à un comportement, une entrée E_{C_i} est *reconnue* pour une règle R d'antécédents $a_1, a_2, ...a_n$ et de confiance $conf_R$ si et seulement si il existe $i_1 < i_2 < \ldots < i_n$ des entiers tels que $a_1 \subseteq u_{i_1}$, $a_2 \subseteq u_{i_2}, \ldots$ $a_n \subseteq u_{i_n}$ et $conf_R > conf$.

Lorsqu'une entrée est jugée suffisamment significative par le gestionnaire de correspondance, la page demandée par l'utilisateur est modifiée par le gestionnaire de pages en ajoutant dynamiquement les liens dynamiques vers les conséquents de la règle reconnue. L'applet recupère alors l'URL d'accès à cette page et l'affiche sur le navigateur. Si aucune règle ne correspond au comportement du client, l'URL vers la page demandée est retournée à l'applet pour qu'elle puisse l'afficher.

Exemple 6 Dans les différentes règles obtenues à partir du fichier access log de l'IUT, nous avons remarqué que 85% des utilisateurs (*confiance* de la règle) qui accèdent au programme du Département Informatique, après être passés par la page de présentation générale de l'IUT et celle du Département, interrogent ensuite le serveur sur les débouchés possibles avec un DUT après être repassés par la présentation du département comme l'illustre le schéma 12.

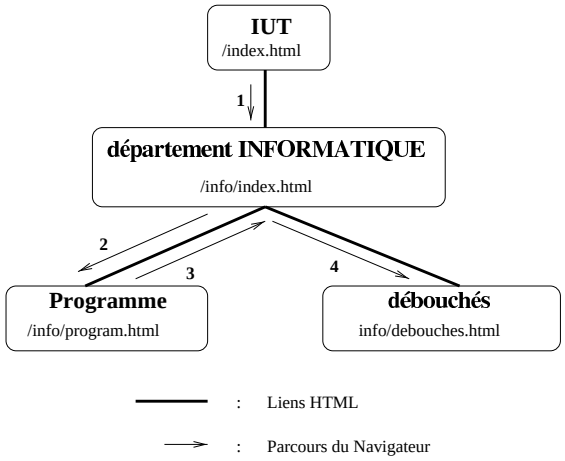

FIG. 12 – *Un exemple de parcours navigationnel*

Considérons un client qui au cours de sa navigation accède aux pages *<(index.html) (info/genera.html) (info/program.html)>*, cette séquence est reconnue dans la base de règles par la règle précédente de confiance 85%. Si cette confiance est supérieure au paramètre passé par l'utilisateur, un lien correspondant à chaque conséquent de cette règle est ajouté à la page. Dans notre cas, un lien vers la page "débouchés" est dynamiquement inséré dans l'URL concernant le programme comme l'illustre la figure 13.
La figure 14 illustre une partie de la hiérarchie des pages HTML mise en place ainsi que la position des applets. Chaque pointeur, dans une page, référence une URL contenant l'applet et chaque applet peut pointer vers une et une seule page html (dynamique ou non).

D'après les expériences réalisées, l'utilisation d'applet et d'un mécanisme de client/serveur pour afficher la page demandée ne ralentit pas les temps d'accès aux pages (l'accès à une page via une applet met en moyenne moins d'une seconde de plus que l'accès direct aux pages du serveur). Le choix d'utilisation d'applet pour suivre le comportement des utilisateurs a été préféré au mécanisme de *cookies* qui nécessite un accord de l'utilisateur.

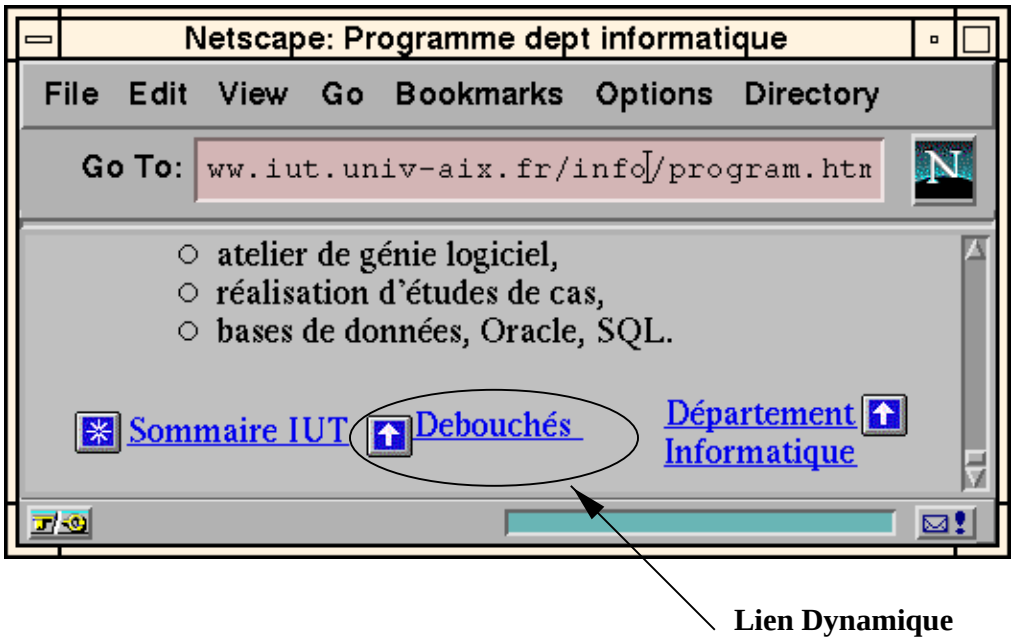

FIG. 13 – *Un exemple de lien dynamiquement ajouté*

A l'heure actuelle, l'insertion d'un lien dynamique par le gestionnaire de page est réalisée par un script Perl. Pour cela chaque page du serveur contient un *tag commentaire* spécial : <!−− *insert URL here*−− > que le script remplace par le lien ajouté. Pour offrir plus de souplesse au système, nous prévoyons de remplacer ce principe par l'utilisation d'un SGBD qui stocke les différentes composantes d'une page ainsi que leur position. Cette approche offre alors l'avantage de pouvoir insérer directement non plus seulement des liens vers d'autres pages mais également des images, des sons ou scripts CGI.

5 Apercu des travaux antérieurs

Une approche pour découvrir des informations à partir de fichiers access log est présentée dans [23, 9]. Les auteurs proposent l'architecture d'un système pour le Web Mining appelée WEBMINER. Par exemple, même si les contraintes de temps ne sont pas prises en compte par le système, une approche pour rechercher des motifs séquentiels est proposée. Dans ce cas, le fichier access log est ré-écrit de manière à regrouper toutes les transactions d'un utilisateur qui sont suffisamment proches dans le temps. Une transaction temporelle est alors vue comme un ensemble d'URL et de temps d'accès tels que les entrées dans le fichier access log sont suffisamment proches pour être regroupées, i.e. elles s'inscrivent dans un intervalle de temps Δt précisé par l'utilisateur. Un algorithme de recherche de règles d'association, similaire à celui de [3], est adapté aux motifs séquentiels. Enfin, le système propose un langage, basé sur SQL, pour offrir un meilleur contrôle sur le processus d'extraction.
Dans [18], un algorithme efficace pour la recherche de séquences d'événements, MINEPI, est utilisé pour extraire des règles à partir du fichier access log de l'Université d'Helsinki. Chaque page consultée est considérée comme un événement et une fenêtre de temps similaire au paramètre Δt de [9] permet de regrouper les entrées suffisamment proches.
Dans le projet WebLogMiner, [33], les auteurs proposent d'utiliser un système OLAP (On-Line Analytical Processing) pour extraire des informations significatives. Les données sont tout d'abord filtrées pour éliminer les informations non pertinentes puis stockées dans une base de données relationnelle. Ensuite, une structure de tableau multidimensionnel, appelée *Web Log data cube*, est construite et chaque dimension représente un champ avec toutes les valeurs possibles décrites par les attributs. Le système OLAP est alors utilisé dans la troisième phase pour appliquer des opérations de *drill-down*,

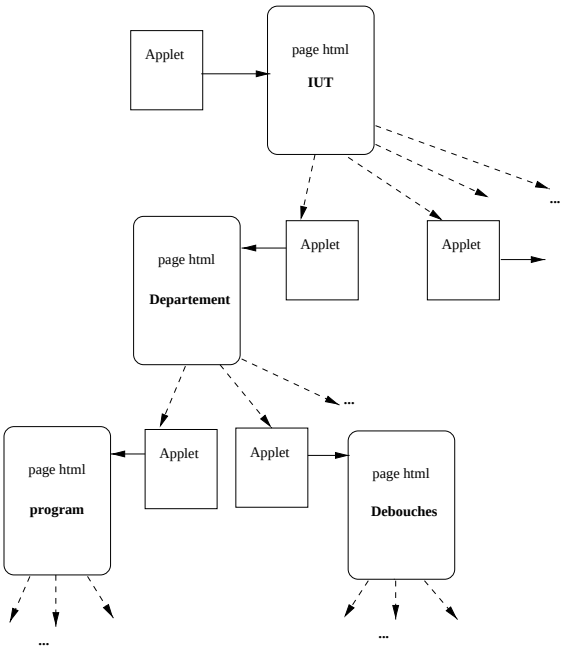

FIG. 14 – *Partie de la hiérarchie des pages*

rool-up, *slice* et *dice* sur le data cube. Ces opérations offrent ainsi la possibilité d'examiner les données sous différents angles. Enfin, des fonctions de data mining comme la caractérisation, la recherche de règles d'association, la prediction ou la classification peuvent être utilisées sur le Web Log data cube. Une approche similaire est présentée dans [10].

6 Conclusion

Dans cet article, nous avons présenté une architecture fonctionnelle d'un système d'extraction de connaissance pour des données stockées par un serveur Web. Les tests sur des fichiers access log issus de deux origines différentes ont montré que le système était capable d'extraire très rapidement, via l'algorithme PSP, les informations contenus dans les fichiers. En outre, les expériences sur la modification dynamique de l'organisation hypertexte du serveur Web de l'IUT ont montré que l'approche retenue était très efficace aussi bien en temps de réponse que dans les résultats obtenus.

Nous sommes actuellement en train d'étudier comment améliorer la recherche du comportement des utilisateurs notamment lors de retours en arrière fréquents et comment prendre en compte l'évolution des serveurs.
Une entrée dans le fichier access log est automatiquement ajoutée à chaque fois qu'une requête pour une ressource atteint le serveur. Alors qu'une telle entrée peut refléter l'utilisation des ressources d'un site, elle n'enregistre pas certains comportements de l'utilisateur comme un retour en arrière fréquent ou le rechargement d'une page lorsque les pages sont cachées par le navigateur ou un Proxy. Par exemple, le fait qu'un utilisateur soit obligé de régulièrement revenir en arrière, peut indiquer une mauvaise conception de la navigation du serveur et de telles informations sont importantes pour améliorer la conception du site. Même s'il existe actuellement des solutions basées sur des Applets Java, une topologie du site [29] ou des "client-site" log files [33], elles sont trop contraignantes et nécessitent que l'outil d'extraction puisse avoir accès aux données stockées sur le site du client.

Le deuxième problème est vraiment critique dans un contexte de Web usage mining dans la mesure où la taille des fichiers log (acces log, error log, ...) croît très rapidement dans le temps. Dans ce

cadre, certains travaux récents ont montré que l'analyse de telles données pouvait être réalisée par un data warehouse et des techniques OLAP ([10, 33]). Cependant, il semble intéressant de proposer une approche incrémentale qui offrirait d'une part d'utiliser les résultats obtenus par les processus d'extraction précédents afin de réduire le coût de recherche dans la base mise à jour et d'autre part de prendre directement en compte les modifications sur la structure d'un site Web.

Références

[1] S. Abiteboul, D. Quass, J. McHugh, J. Widom, and J.L Wiener. The Lorel Query Language for Semi-Structured Data. *International Journal on Digital Libraries*, 1(1):68–88, April 1997.

[2] R. Agrawal, T. Imielinski, and A. Swami. Mining Association Rules between Sets of Items in Large Databases. In *Proceedings of the 1993 ACM SIGMOD Conference*, pages 207–216, Washington DC, USA, May 1993.

[3] R. Agrawal and R. Srikant. Fast Algorithms for Mining Generalized Association Rules. In *Proceedings of the 20th International Conference on Very Large Databases (VLDB'94)*, Santiago, Chile, September 1994.

[4] R. Agrawal and R. Srikant. Mining Sequential Patterns. In *Proceedings of the 11th International Conference on Data Engineering (ICDE'95)*, Tapei, Taiwan, March 1995.

[5] S. Brin, R. Motwani, J.D. Ullman, and S. Tsur. Dynamic Itemset Counting and Implication Rules for Market Basket Data. In *Proceedings of the International Conference on Management of Data (SIGMOD'97)*, pages 255–264, Tucson, Arizona, May 1997.

[6] S. Chawathe, H. Garcia-Molina, J. Hammer, K. Ireland, Y. Papakonstantinou, J. Ullman, and J. Widom. The TSIMMIS Project: Integration of Heterogeneous Information Sources. In *Proceedings of the IPSJ Conference*, pages 7–18, Tokyo, Japan, October 1994.

[7] D.W. Cheung, B. Kao, and J. Lee. Discovering User Access Patterns on the World-Wide Web. In *Proceedings of the 1st Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'97)*, February 1997.

[8] World Wide Web Consortium. httpd-log files. In *http://lists.w3.org/Archives*, 1998.

[9] R. Cooley, B. Mobasher, and J. Srivastava. Web Mining: Information and Pattern Discovery on the World Wide Web. In *Proceedings of the 9th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'97)*, November 1997.

[10] C. Dyreson. Using an Incomplete Data Cube as a Summary Data Sieve. *Bulletin of the IEEE Technical Committee on Data Engineering*, pages 19–26, March 1997.

[11] U.M. Fayad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors. *Advances in Knowledge Discovery and Data Mining.* AAAI Press, Menlo Park, CA, 1996.

[12] M. Fernández, D. Florescu, J. Kang, and A. Levy. Catching the Boat with Strudel: Experiences with a Web-Site Management System. *Proceedings of the International Conference on Management of Data (SIGMOD'98) - SIGMOD record*, 27(2):414–425, 1998.

[13] G. Gardarin, P. Pucheral, and F. Wu. Bitmap Based Algorithms For Mining Association Rules. In *Actes des journées Bases de Données Avancées (BDA'98)*, Hammamet, Tunisie, October 1998.

[14] J. Han, Y. Fu, W. Wang, K. Koperski, and O. Zaïane. Dmql: A Data Mining Query Language for Relational Databases. In *Proceedings of the SIGMOD'96 Workshop on Research Issues in Data Mining and Knowledge Discovery (DMKD'96)*, Montreal, Canada, 1996.

[15] HyperNews. Httpd log analyzers. In *http://www.hypernews.org/HyperNews/get/www/log-analyzers.html*, 1998.

[16] C.A. Knoblock, S. Minton, J.L. Ambite, N.Ashish, P.J Modi, I. Musla, A.G. Philpot, and S. Tejada. Modeling Web Sources for Information Integration. In *Proceedings of the 15th National Conference on Artificial Intelligence*, pages 211–218, Madison,Wisconsin, 1998.

[17] H. Lieberman. Letizia: An Agent that Assists Web Browsing. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'95)*, 1995.

[18] H. Mannila, H. Toivonen, and A.I. Verkamo. Discovery of Frequent Episodes in Event Sequences. *Data Mining and Knowledge Discovery*, 1(3), February 1997.

[19] H. Mannila, H. Toivonen, and A. I. Verkano. Discovering Frequent Episodes in Sequences. In *Proceedings of the 1st International Conference on Knowledge Discovery in Databases and Data Mining (KDD'95)*, pages 210–215, Montreal, Canada, August 1995.

[20] F. Masseglia. Le pré-calcul appliqué à l'extraction de sequential patterns en data mining. Technical report, LIRMM, France, June 1998.

[21] F. Masseglia, F. Cathala, and P. Poncelet. The PSP Approach for Mining Sequential Patterns. In *Proceedings of the 2nd European Symposium on Principles of Data Mining and Knowledge Discovery (PKDD'98), LNAI, Vol. 1510*, pages 176–184, Nantes, France, September 1998.

[22] J. McHugh, S. Abiteboul, R. Goldman, D. Quass, and J. Widom. LORE: a Database Management System for Semi-Structured Data. *SIGMOD Record*, 26(3), September 1997.

[23] B. Mobasher, N. Jain, E. Han, and J. Srivastava. Web Mining: Pattern Discovery from World Wide Web Transactions. Technical Report TR-96-050, Department of Computer Science, University of Minnesota, 1996.

[24] L. Moreau and N. Gray. A Community of Agents Maintening Link Integrity in the World-Wide Web. In *Proceedings of the 3rd International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology (PAAM'98)*, pages 221–233, London, UK, March 1998.

[25] A. Mueller. Fast Sequential and Parallel Algorithms for Association Rules Mining: A Comparison. Technical Report CS-TR-3515, Department of Computer Science, University of Maryland-College Park, August 1995.

[26] C. Neuss and J. Vromas. *Applications CGI en Perl pour les Webmasters*. Thomson Publishing, 1996.

[27] N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Efficient Mining of Association Rules Using Closed Itemset Lattices. *Information Systems*, 19(4):33–54, 1998.

[28] M. Pazzani, J. Muramatsu, and D. Billsus. Syskill and Webert: Indentifying Interesting Web Sites. In *Proceedings of the AAAI Spring Symposium on Machine Learning In Information Access*, Portland, Oregon, 1996.

[29] J. Pitkow. In Search of Reliable Usage Data on the WWW. In *Proceedings of the 6th International World Wide Web Conference*, pages 451–463, Santa Clara, CA, 1997.

[30] A. Savasere, E. Omiecinski, and S. Navathe. An Efficient Algorithm for Mining Association Rules in Large Databases. In *Proceedings of the 21 st International Conference on Very Large Databases (VLDB'95)*, pages 432–444, Zurich, Switzerland, September 1995.

[31] R. Srikant and R. Agrawal. Mining Sequential Patterns: Generalizations and Performance Improvements. In *Proceedings of the 5th International Conference on Extending Database Technology (EDBT'96)*, pages 3–17, Avignon, France, September 1996.

[32] H. Toivonen. Sampling Large Databases for Association Rules. In *Proceedings of the 22nd International Conference on Very Large Databases (VLDB'96)*, September 1996.

[33] 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 on Advances in Digital Libraries Conference (ADL'98)*, Santa Barbara, CA, April 1998.