# Web Usage Mining Inter-Sites : Analyse du Comportement des Utilisateurs à Impact Immédiat

## Intersites Web Usage Mining: Users Behaviour Analysis with Immediate Impact

F. Masseglia[1,2] M. Teisseire[1] P. Poncelet[1]

[1]LIRMM UMR CNRS 5506, 161 Rue Ada, 34392 Montpellier Cedex 5, France
[2]Laboratoire PRiSM, Université de Versailles, 45 Avenue des Etats-Unis, 78035 Versailles Cedex, France

E-mail: {massegli, teisseire, poncelet}@lirmm.fr

**Abstract**

Analyzing access log files is usually called Web Usage Mining and provides very useful knowledge for performance enhancement, restructuring a Web site and customer targeting in electronic commerce. These techniques are however limited to a single Web server log file analysis (Intrasite Web Usage Mining). We propose in this paper the Web Usage Mining Inter-Sites notion in order to study the behaviours of users shared by several partner Web sites. The provided knowledge is thus used to dynamically improve the hypertext organization of a Web site, with two main advantages over the classic approach: more accurate frequent behaviours are found and the result allows an immediate impact. Finally, we prove that there exists a correlation between the properties of the frequent patterns, at the intersection of the considered data, and the user's minimum support. Through this study, we are able to prove that a reduction from the Web Usage Mining Inter-Sites problem to the incremental data mining problem, allows the knowledge discovery phase to be optimized.
**Keywords:** data mining, web usage mining, sequential patterns, incremental, intersites.

**Résumé**

L'analyse des fichiers access log, appelée Web Usage Mining, offre des informations très utiles pour améliorer les performances du réseau, restructurer un site et surtout cibler le comportement des clients dans le cadre du commerce électronique. Ces techniques sont cependant limitées à l'analyse d'un seul fichier log (Web Usage Mining Intra-Site), issu d'un serveur Web. Nous proposons dans cet article la notion de Web Usage Mining Inter-Sites, qui consiste à analyser le comportement des utilisateurs communs à plusieurs sites Web partenaires. Les connaissances obtenues sont alors utilisées pour optimiser dynamiquement l'organisation hypertexte d'un serveur avec deux avantages majeurs sur l'approche classique : un niveau de précision accru et un effet immédiat des résultats obtenus. Enfin, nous prouvons l'existence d'une corrélation entre les propriétés des fréquents, à l'intersection des données mises en jeu, et le support minimum. Par cette analyse, nous pouvons démontrer que le problème du Web Usage Mining Inter-Sites admet une réduction au problème du Data Mining Incrémental, optimisant ainsi la phase de recherche de connaissances.

**mots clés :** data mining, web usage mining, motifs séquentiels, incrémental, inter-sites.

## 1 Introduction

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'extraction de connaissances à partir de ces fichiers, est

généralement appelée *Web Usage Mining*. Les techniques impliquées dans ce domaine s'intéressent au problème de la recherche de motifs comportementaux des utilisateurs à partir d'un serveur Web afin d'extraire des relations entre les données stockées [13, 19, 6, 17, 7, 10]. Les méthodes de Data Mining utilisées dans ce cadre sont celles destinées à trouver les motifs les plus fréquents, répartis parmis les entrées dans la base de données et permettent la découverte de *règles d'association* ou de *motifs séquentiels*. Bien que la problématique des règles d'association permette de découvrir des corrélations très utiles pour des besoins de type marketing, elle n'est pas optimale pour analyser le comportements d'utilisateurs d'un site Web. En effet, l'absence d'ordre entre les éléments fréquents est un facteur qui empêche de savoir comment s'organisent les demandes fréquentes d'URL et donc de prévoir les comportements futurs. Pour apporter la notion de temporalité dans les motifs extraits, la notion de motifs séquentiels ([2, 18]) propose d'estampiller chaque transaction et d'extraire des motifs ordonnés dans le temps. L'extraction de motifs séquentiels, depuis un fichier access log, permet 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 2 jours suivants, l'URL* `/jdk1.1.6/docs/relnotes/deprecatedlist.html`

Ces 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.

Cependant, nous pouvons constater qu'à l'heure actuelle le nombre de partenariats entre sites ne cesse d'augmenter. Les internautes se voient alors offerte la possibilité de naviguer d'un site à l'autre et les fichiers access log des partenaires mis en jeu, ont en commun des visiteurs qui ne sont pas dûs au hasard, mais à ces redirections tacites.

Prenons le bref exemple d'un site de vente en ligne $SV$, qui propose des articles très variés. Ces articles viennent de fabricants qui possèdent, chacun, un site sur le Web : $SF1$, $SF2$, ..., $SFn$. Chacun de ces sites propose la description de ses produits (entre autres informations) et des liens vers les sites de revendeurs proposant ses articles (dont $SV$). Le visiteur d'un site $SFx$ a alors de fortes chances d'être redirigé vers $SV$ s'il décide de passer à l'achat. Pour $SV$, tenir compte de l'historique de ce visiteur et des comportements fréquents de ceux qui, avant lui, ont suivi cette redirection, présente sans nul doute des intérêts que la seule analyse du fichier access log de $SV$ ne suffit pas à satisfaire.

Dans cet article, nous proposons, dans un premier temps, d'apporter une dimension supplémentaire au Web Usage Mining classique (Intra-Site). Dans un deuxième temps, nous présentons une étude visant à mettre en corrélation les propriétés des fréquents, à l'intersection des données mises en jeu, et le support minimum.

Le premier aspect de notre contribution propose de définir le concept de Web Usage Mining Inter-Sites. Dans ce cadre, les motifs recherchés ont une portée plus grande en comparaison au Web Usage Mining Intra-Site. Ils permettent de découvrir des comportements fréquents pour une population ciblée, correspondant aux visiteurs communs à deux sites Web. Le fait de restreindre la phase d'extraction de connaissances sur cette partie de la population répond à un objectif déterminant du processus de KDD : la sélection, étape du prétraitement des données. Du point de vue du Web Usage Mining Inter-Sites, les visiteurs sont naturellement indentifiés comme appartenant à la catégorie que l'on va étudier. Dans notre exemple, cette catégorie est constituée de l'intersection entre l'ensemble des visiteurs du site $SFx$ d'une part, et ceux de $SV$ d'autre part.

Pour le site $SV$, qui veut connaître le comportement des visiteurs que lui envoie $SF1$, les motifs séquentiels découverts dans le cadre du Web Usage Mining Inter-Sites sont alors de la forme suivante :

*70 % des clients envoyés par SF1, ont visité, de façon séquentielle, les pages suivantes :*
`SF1(/téléviseurs/home-cinéma/cat1/modèles.html)`
`SF1(/téléviseurs/home-cinéma/cat1/modèle3/revendeur1.html)`
`SV(/magasin/hi-fi/home-cinéma/SF1/tarif-modèle3.html)`
`SV(/magasin/hi-fi/home-cinéma/SF1/tarif-modèle4.html)`
`SV(/magasin/hi-fi/enceintes/SF1/modeles.html)`
`SV(/magasin/hi-fi/enceintes/SF3/tarif-modèle8.html)`

Les avantages se situent alors :

- dans le potentiel offert par ces résultats : si $SV$ veut modifier l'organisation hypertexte pour un client venu de SF1 et dont le comportement est similaire à ce comportement fréquent, alors cette modification peut prendre effet dès l'arrivée de ce client sur le site $SV$ (contrairement aux techniques similaires dans le cadre du Web Usage Mining Intra-Site).
- dans la confiance avec laquelle ces résultats sont calculés : en effet tous les visiteurs ne sont pas pris en compte dans la phase d'extraction, donc les résultats obtenus sont dédiés aux visiteurs de la catégorie qui préoccupe $SV$ (i.e. leur confiance est supérieure).

Le deuxième aspect de notre contribution prouve que la phase d'extraction de connaissances, mettant en jeu l'intersection des données provenant des deux sites, ne doit pas nécessairement être envisagée comme un processus classique. En effet, les propriétés que nous utilisons nous permettent de conclure sur la possibilité d'employer un algorithme de data mining incrémental, dont les gains, en termes de temps de calcul, ont été mis en relief dans [15, 4, 3, 9, 11].

L'article est organisé de la manière suivante. Le paragraphe 2 propose un exposé de la problématique concernant la recherche de motifs séquentiels dans les bases de données. Une fois ces concepts établis, le paragraphe 3 en explique leur utilisation dans le contexte du Web Usage Mining. Le type de résultats obtenus et la façon dont ils peuvent être exploités y sont également abordés. Notre contribution se place tout d'abord sur la mise au point d'une nouvelle problématique : le Web Usage Mining Inter-Sites. Cette problématique est exposée dans le paragraphe 4. Ce paragraphe détaille aussi la manière dont nous proposons de résoudre ce problème grâce à l'utilisation d'un algorithme d'extraction incrémentale de motifs séquentiels. Enfin dans le paragraphe 5, nous concluons en évoquant les possibilités futures relatives à ce travail.

## 2 Définitions préliminaires

Ce paragraphe expose et illustre la problématique liée à l'extraction de motifs séquentiels dans de grandes bases de données. Il reprend les différentes définitions proposées dans [1] et [2].

Dans [1], 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 implication 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 [2] 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 $T_1$, $T_2$, ..., $T_n$ 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".*

**Définition 3** Le *support* de $s$, noté $supp(s)$, est le pourcentage de toutes les séquences dans $D$ qui supportent (contiennent) $s$. Si $supp(s) \geq minsupp$, avec une valeur de support minimum *minsupp* fixée par l'utilisateur, la séquence $s$ est dite *fréquente*.

# 3 Web Usage Mining Intra-site

Dans [10], nous proposons une méthode de Web Usage Mining basée sur une architecture Intra-site. Ce paragraphe propose de reprendre les concepts essentiels de ces travaux, afin de présenter de façon synthétique, les procédés mis en œuvre lors de l'analyse du comportement des utilisateurs d'un site Web.

## 3.1 Adapter la problématique des Motifs Séquentiels

Les principes généraux sont similaires à ceux du processus d'extraction de connaissances exposés dans [8]. 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.
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*). Le fichiers access log peuvent varier selon les systèmes qui hébergent le serveur, mais présentent tous en commun trois champs : l'adresse du demandeur, l'URL demandée et la date à laquelle cette demande a eu lieu. Parmi ces différents types de fichiers, nous avons retenu

dans cet article le format spécifié par le CERN et la NCSA [5], une entrée contient des enregistrements formés de 7 champs séparés par des espaces [14] :

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

La figure 1 illustre un extrait de fichier access log du serveur Web du Lirmm.

```
193.49.105.224 - - [29/Nov/2000:18:02:26 +0200] "GET /index.html HTTP/1.0 " 200 1159
193.49.105.224 - - [29/Nov/2000:18:02:27 +0200] "GET /PtitLirmm.gif HTTP/1. 0" 200 1137
193.49.105.224 - - [29/Nov/2000:18:02:28 +0200] "GET /acceuil_fr.html HTTP/1. 0" 200 1150
193.49.105.224 - - [29/Nov/2000:18:02:30 +0200] "GET /venir/venir.html HTTP/1.0" 200 1141
193.49.105.225 - - [29/Oct/2000:18:03:07 +0200] "GET /index.html HTTP/1.0" 200 1051
193.49.105.225 - - [16/Oct/2000:20:34:32 +0100] "GET /form/formation.html HTTP/1.0" 200 14617
193.49.105.225 - - [31/Oct/2000:01:17:40 +0200] "GET /form/formation.html#doct HTTP/1.0" 304 -
193.49.105.225 - - [31/Oct/2000:01:17:42 +0200] "GET /form/theses2000.html HTTP/1.0" 304 -
193.49.105.56 - - [22/Nov/2000:11:06:11 +0200] "GET /lirmm/bili/ HTTP/1.0" 200 4280
193.49.105.56 - - [22/Nov/2000:11:06:12 +0200] "GET /lirmm/bili/rev_fr.html HTTP/1.0" 200 2002
193.49.105.56 - - [07/Dec/2000:11:44:15 +0200] "GET /ress/ressources.html HTTP/1.0" 200 5003
```

Figure 1: Exemple de fichier access log

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” est réalisée. Au cours de la phase de tri et afin de rendre plus efficace le traitement de l'extraction de données, 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.

**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 $1 \leq k \leq m$, $l_k^g.URL$ représente l'objet demandé par le client $g$ à la date $l_k^g.time$, et pour tout $1 \leq j < k$, $l_k^g.time > l_j^g.time$.

La figure 2 illustre un exemple de fichier obtenu après la phase de pré-traitement. A chaque client correspond une suite de “dates” (événements) et la traduction de l'URL demandée par ce client à cette date.

| Client | d1 | d2 | d3 | d4 | d5 |
|---|---|---|---|---|---|
| 1 | 10 | 30 | 40 | 20 | 30 |
| 2 | 10 | 30 | 60 | 20 | 50 |
| 3 | 10 | 70 | 30 | 20 | 30 |

Figure 2: Exemple de fichier résultat issu de la phase de pré-traitement

L'objectif est alors de déterminer, gràce à une phase d'extraction, les séquences de ce jeu de données, qui peuvent être considérées comme fréquentes selon la définition 3. Les résultats obtenus sont du type `< ( 10 ) ( 30 ) ( 20 ) ( 30 ) >` (ici avec un support minimum de 100% et en appliquant les algorithmes de fouille de données sur le fichier représenté par la figure 2). Ce dernier résultat, une fois re-traduit en termes d'URL, confirme la découverte d'un comportement commun à $minSup$ utilisateurs et fournit l'enchaînement des pages qui constituent ce comportement fréquent. Toutes ces étapes (pré-traitement des données, data mining, exploitation des résultats...) sont intégrées dans la plateforme WebTool, dont la figure 3 illustre un exemple d'utilisation.

### 3.2 Modification dynamique des liens hypertexte

Conjointement au système WebTool, nous avons développé un générateur de liens dynamiques dans des pages Web à partir des règles obtenues lors du processus d'extraction [12]. Le but du générateur est

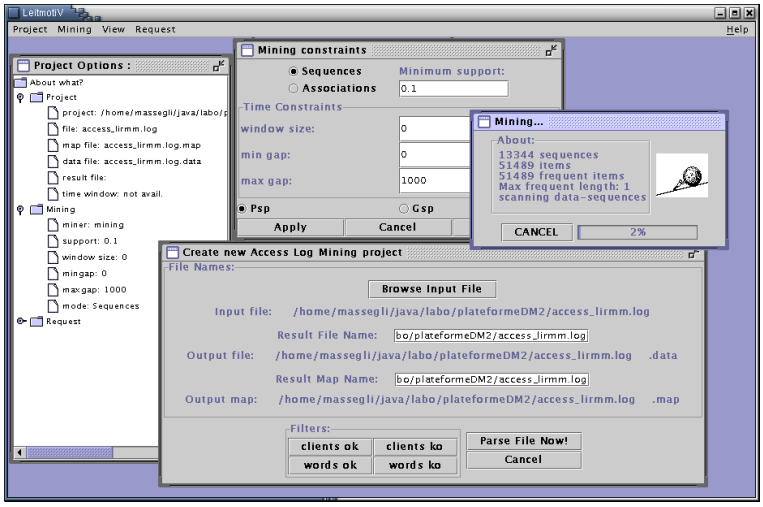


Figure 3: WebTool lors de l'extraction de motifs séquentiels

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. L'architecture fonctionnelle de l'approche est décrite dans la figure 4.

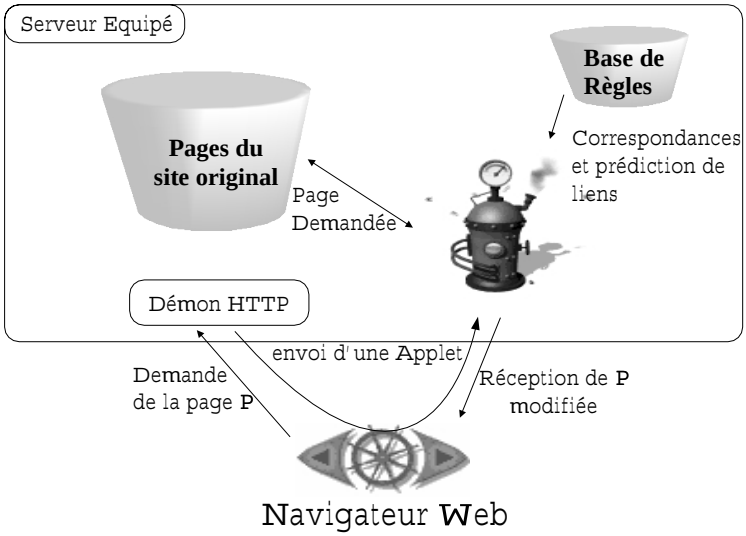


Figure 4: Architecture générale

Le serveur Web (démon *http*) réagit à l'envoi d'une requête par un client en lui retournant une page contenant une applet. 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.
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 2** Dans les différentes règles obtenues à partir du fichier access log du Lirmm, nous avons remarqué que 65% des utilisateurs (*confiance* de la règle) qui accèdent à la présentation du Lirmm, après être passés par la page d'accueil, reviennent sur cette page d'accueil, avant de demander la partie du site concernant les formations. Une fois la page des formations consultées ils demandent la page concernant la formation doctorale et enfin celle du DEA informatique comme l'illustre le schéma 5.

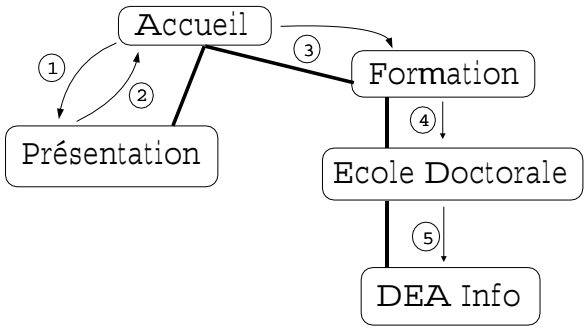


Figure 5: Un exemple de parcours navigationnel

Considérons un client qui au cours de sa navigation accède aux pages *<(accueil) (presentation) (accueil) (formation)>*. Le début de cette séquence est reconnu dans la base de règles par la règle précédente de confiance 65%. 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 "DEA Info" est dynamiquement inséré dans l'URL concernant les formations, aux côtés de celui qui mène à la page concernant l'école doctorale.

# 4 Web Usage Mining Inter-sites

Ce paragraphe vise deux objectifs. Tout d'abord nous proposons de définir plus formellement le concept de Web Usage Mining Inter-Sites et de présenter la conjoncture motivant cette approche ainsi que les intérêts mis en jeu par cette nouvelle notion. Une fois le problème posé, nous présentons une solution basée sur l'utilisation d'un algorithme de data mining incrémental. Le fonctionnement de cette solution repose sur la corrélation, que nous avons mise en évidence, entre le support minimum et les propriétés des fréquents.

## 4.1 Présentation et motivations

Une des caractéristiques incontournables de la structure du Web à l'heure actuelle, réside dans les possibilités qu'offrent les sites de passer de l'un à l'autre quand leurs centres d'intérêts le justifient. Ce type de liens, qui représente ce que l'on peut qualifier de partenariat entre sites web, apporte un nouvel élément qui se traduit au niveau des fichiers access log : le nombre d'utilisateurs communs aux sites qui pratiquent le partenariat est croissant. De la même façon que le cumul des données a créé le besoin de méthodes pour le data mining, ce phénomène de partenariat en en train de créer le besoin de méthodes pour le Web Usage Mining Inter-Sites.

**Le partenariat**

Reprenons l'exemple de partenariat décrit en introduction. L'exemple 3 illustre la façon dont les fichiers access logs des sites Web mis en jeu peuvent être fusionnés.

**Exemple 3** *La figure 6 illustre la façon dont les clients sont sélectionnés, à partir des fichiers access log de $SF1$ et $SV$, dans le but de construire le fichier access log qui leur est commun. Ce nouveau fichier "Log+" contient les entrées correspondant aux visiteurs de $SF1$ qui ont utilisé un lien de $SF1$ vers $SV$.*

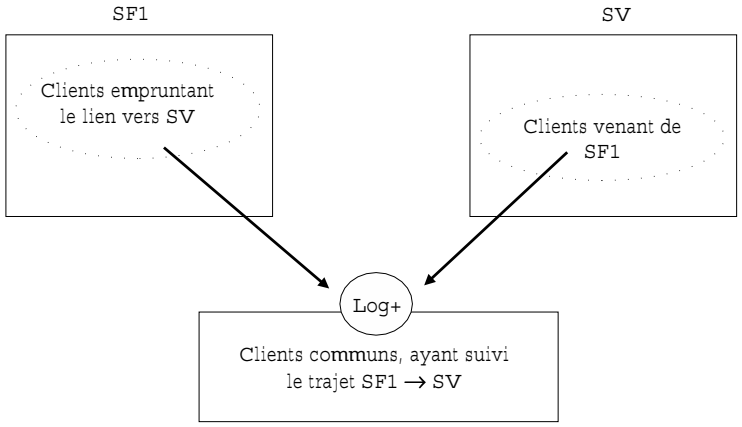


Figure 6: Un exemple de mise en commun des fichiers access log

Cette nécessité de mettre en commun les fichiers access log, nous conduit dans un premier temps à préciser la notion de *log* d'abord introduite par la définition 4 :

| Site | URL | Date |
|---|---|---|
| $SF1$ | a | 01/01/2001 15h30:25s |
| $SF1$ | b | 01/01/2001 15h30:28s |
| $SF1$ | To_SV | 01/01/2001 15h31:10s |
| $SV$ | From_SF1 | 01/01/2001 15h31:11s |
| $SV$ | d | 01/01/2001 15h31:50s |
| $SF1$ | a | 03/01/2001 16h00:25s |
| $SF1$ | c | 03/01/2001 16h00:28s |
| $SF1$ | To_SV | 03/01/2001 16h01:10s |
| $SV$ | From_SF1 | 03/01/2001 16h01:11s |
| $SV$ | e | 03/01/2001 16h01:50s |

Figure 7: Historique du client $C1$, sans le concept de session

**Définition 5** Soit *Log* un ensemble d'entrées dans le fichier access log. Une entrée $g$, $g \in Log$, est un tuple $g =< ip_g.s, \{(l_1^g.URL, l_1^g.time), ..., (l_m^g.URL, l_m^g.time)\} >$ tel que pour $1 \leq k \leq m$, $l_k^g.URL$ représente l'objet demandé par le client $g$ à la date $l_k^g.time$, et pour tout $1 \leq j \leq m / k > j$, $l_k^g.time > l_j^g.time$. $s$, qui représente le numéro de session de la séquence de navigation de $g$, est déterminé par l'algorithme 1.

Log $SF1$

| Client | URL | Date |
|---|---|---|
| $C1$ | a | 01/01/2001 15h30:25s |
| $C1$ | b | 01/01/2001 15h30:28s |
| $C1$ | To_SV | 01/01/2001 15h31:10s |
| $C1$ | a | 03/01/2001 16h00:25s |
| $C1$ | c | 03/01/2001 16h00:28s |
| $C1$ | To_SV | 03/01/2001 16h01:10s |

Log $SV$

| Client | URL | Date |
|---|---|---|
| $C1$ | From_SF1 | 01/01/2001 15h31:11s |
| $C1$ | d | 01/01/2001 15h31:50s |
| $C1$ | From_SF1 | 03/01/2001 16h01:11s |
| $C1$ | e | 03/01/2001 16h01:50s |

Figure 8: access log des sites $SF1$ et $SV$, sans concept de session

Log $SF1$

| Client | URL | Date |
|---|---|---|
| $C1.1$ | a | 01/01/2001 15h30:25s |
| $C1.1$ | b | 01/01/2001 15h30:28s |
| $C1.1$ | To_SV | 01/01/2001 15h31:10s |
| $C1.2$ | a | 03/01/2001 16h00:25s |
| $C1.2$ | c | 03/01/2001 16h00:28s |
| $C1.2$ | To_SV | 03/01/2001 16h01:10s |

Log $SV$

| Client | URL | Date |
|---|---|---|
| $C1.1$ | From_SF1 | 01/01/2001 15h31:11s |
| $C1.1$ | d | 01/01/2001 15h31:50s |
| $C1.2$ | From_SF1 | 03/01/2001 16h01:11s |
| $C1.2$ | e | 03/01/2001 16h01:50s |

Figure 9: access log des sites $SF1$ et $SV$, avec concept de session

Dans cette définition, la différence réside dans l'ajout d'un numéro de session $s$, associé au client. Ce numéro de session permet de décliner le client en plusieurs versions, en fonction du nombre de ses visites sur le site. Il est déterminé par un délai maximum, fixé par l'utilisateur, qui permet de décider si un client est encore dans un même objectif de navigation, ou si il s'est reconnecté sur le site après un temps $N$. Ce temps fixé par l'utilisateur, si il est dépassé entre deux actions du client (par exemple deux actions séparées par un intervalle de 2 jours), permet de conclure que ce client peut être considéré comme une nouvelle entrée dans le fichier access log, afin d'éviter des confusions comme celle décrite dans l'exemple 4.

**Exemple 4** *La figure 7 illustre un cas de figure sans la notion de session. On peut y constater que le client $C1$ s'est connecté sur le site $SF1$ à deux jours d'intervalle et en suivant le lien vers $SV$ systématiquement. La séquence de navigation de $C1$, d'après les fichiers logs illustrés par la figure 8, devient alors* `<(a) (b) (To_SV) (a) (c) (To_SV) >`*, sur $SF1$ et* `<(From_SF1) (d) (From_SF1) (e)>` *sur $SV$. Ces séquences présentent sans doute un intérêt pour les sites observés indépendamment, mais leur niveau de détail est insuffisant pour garder une trace historisée des passages de $C1$ entre $SF1$ et $SV$. Grâce à la notion de session décrite par la définition 5, les entrées dans le fichier log de $SF1$ distinguent les deux types de navigation de $C1$, en déclinant deux versions de cet utilisateur (i.e. $C1.1$ et $C1.2$) qui seront alors considérées comme deux clients différents. Cette version, exploitable pour le Web Usage Mining Inter-Sites, des fichiers access log est illustrée par la figure 9, et correspond au même historique qui, sans ce concept, serait problématique. Le client $C1$ se voit donc décliné en deux versions ($C1.1$ et $C1.2$) et les séquences de navigation de ces nouveaux clients sur $SF1$ et $SV$ sont alors :*

- *pour $C1.1$ :* `<(a) (b) (To_SV)>` *sur $SF1$ et* `<(From_SF1) (d)>` *sur $SV$.*
- *pour $C1.2$ :* `<(a) (c) (To_SV)>` *sur $SF1$ et* `<(From_SF1) (e)>` *sur $SV$.*

Pour terminer la présentation de cette notion de session, précisons que le fichier access log sans sessions de $SFx$, peut être retrouvé sans difficulté en supprimant le $.s$ dans le fichier. Ainsi $C1.1$ et $C1.2$ (pour reprendre l'access log de $SF1$ dans l'exemple 4) deviennent à nouveau $C1$.

La définition 6, présente le concept de $Log+$, qui regroupe pour deux sites, les visiteurs qu'ils ont en commun (i.e. le fichier "$Log+$" de la figure 6).

**Définition 6** Soit $Log_A$ et $Log_B$ l'ensemble des entrées des fichier log des sites $A$ et $B$. Soit $Log+$ le fichier log crée à partir de $Log_A$ et $Log_B$. Une entrée $g$, $g \in Log+$, est un tuple $g =< ip_g.s, \{(l_1^g.URL, l_1^g.time), ..., (l_m^g.URL, l_m^g.time)\} >$ respectant les propriétés suivantes :

- $\exists\ i,\ 1 \leq i \leq m /\ l_i^g.URL$=To_B. Si $i < m$ alors $(l_{i+1}^g.URL)$=From_A et $\forall z > i+1 /\ (l_z^g.URL) \neq$From_A.
- $\forall g_A \in Log_A$ si $l_m^{g_A}.URL$ =To_B, avec $m$ la longueur de $g_A$, alors $\exists g \in Log+ / \forall i \in (1..m)\ l_i^{g_A} = l_i^g$
- $\forall g_B \in Log_B$ si $l_1^{g_A}.URL$ =From_A, avec $m$ la longueur de $g_B$, alors $\exists g \in Log+$ de longueur $n$ / $\forall i \in ((n-m)..n)\ l_i^{g_B} = l_i^g$

Le premier point de la définition 6 exprime le fait que toutes les entrées dans $Log+$ représentent la navigation d'un utilisateur qui est passé du site $A$ à son site partenaire $B$ en utilisant les liens du partenariat. Ce premier point montre aussi que si un échec est survenu pendant l'acheminement d'un client du site $A$ vers le site $B$ (pour cause de lien "cassé", problème réseau, sécurité, ou autre...) l'entrée est conservé dans le fichier $Log+$ car elle pourra permettre de déterminer l'importance, les causes et les solutions à envisager à ces failles dans les redirections. Enfin, il garantit que pour un client associé à une session, il ne peut exister, au plus, qu'une seule demande pour l'objet $From\_A$. Le second point assure que si un utilisateur du site $A$ a emprunté un tel lien, alors il est le début d'une entrée dans le fichier $Log+$. Le troisième point assure que si un utilisateur du site $B$ est arrivé grâce à un lien du partenariat, alors il termine une entrée de $Log+$. Finalement, le fichier $Log+$ peut être considéré comme le fichier access log du site virtuel constitué par $A \cap B$ restreint aux clients qui sont passés de $A$ à $B$ en suivant les liens du partenariat.

Exprimons les navigations des clients illustrées par la figure 9 sour forme de séquences de navigation. Le client $C1.1$, au cours de sa session, a réalisé le parcours suivant : `<(a) (b) (To_SV) (From_SF1) (d)>`. Le client $C1.2$, aura le parcours : `<(a) (c) (To_SV) (From_SF1) (e)>`. Ces deux séquences, associées aux clients qui les supportent et aux dates auxquelles les URLs sont demandées, forment les deux entrées du fichier $Log+$ correspondant aux fichiers log de $SF1$ et $SV$. Notons que sans le concept de session, ce fichier $Log+$ serait composé d'une seule entrée : `<(a) (b) (To_SV) (a) (c) (To_SV) (From_SF1) (d) (From_SF1) (e)>` et que cette séquence ne nous permet pas, directement, de fournir des connaissances sûres, dans la mesure où les différents passages de $SF1$ à $SV$ sont "brouillés".

### Une qualité de connaissance accrue

La séquence de navigation d'un utilisateur $C1$, commun à deux sites $A$ et $B$, selon la définition 6, est de la forme : `<`$s1$`=Navigation sur `$A$, $s2$`=Navigation sur `$B$`>`. Il s'agit en fait de la concaténation de deux séquences de navigation $s1$ et $s2$, enregistrées sur les fichiers access log de $A$ et $B$. De plus, $s2$ correspond à la suite directe des opérations faites par $C1$ sur $B$, après avoir navigué selon le schéma exprimé par $s1$ sur $A$ (i.e. $s2$ représente les opérations faites par $C1$ sur $B$, après que le dernier objet de $s1$ ait redirigé $C1$ vers le premier objet de $s2$).
Lorsque l'on filtre les données inscrites dans le fichier access log de $B$, afin de ne garder que celles qui sont issues de navigations qui suivent le partenariat entre $A$ et $B$, le fichier considéré ne concerne

**function** *accepterRequête*
Input: Un client $Cn.x$, dans sa session $x$, demandant un objet sur le site $A$ ou $B$.
$N$ la durée spécifiée par l'utilisateur pour la durée d'une session.
Output: $Cn.y$ le client avec un éventuel nouveau numéro de session.
**if** *Cn demande From_ A sur B* **then**
// $Cn$ passe du site $A$ au site partenaire $B$ par un lien de partenariat
$y = x$; // On ne change rien, $Cn$ garde son numéro de session;
**endif**
**return**($Cn.y$);
**if** *dernierObjet(Cn)* $\in$ $B$ **and** *Cn demande une page sur A* **then**
// $Cn$ passe du site $B$ au site partenaire $A$, c'est une nouvelle session
$y = x + 1$; // On incrémente le numéro de session
**endif**
**return**($Cn.y$);
**if** *dateDernierObjet(Cn)*$>N$ **and** *Cn demande une page sur A* **then**
// $Cn$ reste sur $A$ mais sa dernière action a une date plus grande que $N$ : nouvelle session
$y = x + 1$; // On incrémente le numéro de session
**endif**
**return**($Cn.y$);
**end function** *accepterRequête*

Algorithm 1: Mise à jour des numéros de session

qu'une seule classe de la population totale de $B$. A partir de cette classification directe, les résultats que fournit un processus de data mining sur ce sous-ensemble de la population, contiennent des informations présentant une confiance plus élevée, mais vont également révéler des fréquents plus riches. L'exemple 5 donne une illustration de ce phénomène.

**Exemple 5** *Considérons les fichiers access log des sites $SF1$, $SF2$ et $SV$ illustrés par la figure 10. Un processus d'extraction de motifs séquentiels fréquents, avec un support de 100%, appliqué sur le fichier log de $SV$, donne le motif séquentiel suivant :* `<(i) (m) (w)>`. *Considérons à présent le fichier log de $SV$ filtré par la sélection suivante : ne garder que les visiteurs ayant suivi un lien de partenariat entre $SF1$ et $SV$ (i.e. dont la séquence de navigation commmence par* `(From_SF1)`. *On obtient alors un nouveau fréquent* `<(From_SF1) (i) (l) (m) (w)>`. *Ce motif séquentiel fréquent, propose une qualité de connaissance supérieure à celle obtenue dans le cas général, car il est dédié aux seuls visiteurs envoyés par le site $SF1$.*

Les chances de cibler le comportement d'un utilisateur sont alors augmentées dans la mesure où, pour prédire les comportements possibles de cet utilisateur, on considère des fréquents calculés uniquement en fonction de ses prédécesseurs, et non pas les comportements fréquents de tous les utilisateurs. L'utilisation des fréquents ainsi obtenus, sera donc plus efficace grâce à son adéquation avec la catégorie à laquelle appartient le client observé. Les liens proposés par le système de modification dynamique des liens hypertexte, ne seront pas ceux que l'on propose à tout utilisateur en fonction de tous les autres, mais ceux que l'on propose à un utilisateur de la catégorie $\alpha$, en fonction du comportement fréquent $\beta$ des utilisateurs de cette même catégorie.

Toutefois, si l'on veut exploiter ces résultats au cours de la navigation d'un client $C$, il faut procéder à au moins trois étapes (en continuant l'exemple 5) :

1. Accepter la première demande de $C$, à savoir $From\_F1$. Cette page étant la même pour tous les utilisateurs de sa catégorie, on ne peut rien en déduire.

Log $SF1$

| Client | URL | Date |
|---|---|---|
| $C1$ | a | 01/01/2001 15h50:25s |
| $C1$ | b | 01/01/2001 15h50:28s |
| $C1$ | c | 01/01/2001 15h51:10s |
| $C1$ | d | 01/01/2001 16h00:55s |
| $C1$ | To_SV | 01/01/2001 16h01:10s |
| $C2$ | a | 03/01/2001 17h50:25s |
| $C2$ | c | 03/01/2001 17h50:28s |
| $C2$ | d | 03/01/2001 17h51:10s |
| $C2$ | e | 03/01/2001 18h00:55s |
| $C2$ | To_SV | 03/01/2001 18h01:10s |
| $C4$ | u | 04/01/2001 14h30:55s |
| $C4$ | v | 04/01/2001 14h31:10s |

Log $SF2$

| Client | URL | Date |
|---|---|---|
| $C3$ | f | 01/01/2001 15h50:28s |
| $C3$ | g | 01/01/2001 15h51:10s |
| $C3$ | h | 01/01/2001 16h00:55s |
| $C3$ | To_SV | 01/01/2001 16h01:10s |

Log $SV$

| Client | URL | Date |
|---|---|---|
| $C1$ | From_SF1 | 01/01/2001 16h01:11s |
| $C1$ | i | 01/01/2001 16h01:50s |
| $C1$ | j | 01/01/2001 16h11:11s |
| $C1$ | l | 01/01/2001 16h11:50s |
| $C1$ | m | 01/01/2001 16h12:40s |
| $C1$ | w | 01/01/2001 16h12:55s |
| $C2$ | From_SF1 | 03/01/2001 18h01:11s |
| $C2$ | i | 03/01/2001 18h01:50s |
| $C2$ | k | 03/01/2001 18h02:11s |
| $C2$ | l | 03/01/2001 18h02:50s |
| $C2$ | m | 03/01/2001 18h02:53s |
| $C2$ | w | 03/01/2001 18h03:10s |
| $C3$ | From_SF2 | 01/01/2001 16h01:11s |
| $C3$ | i | 01/01/2001 16h01:50s |
| $C3$ | z | 01/01/2001 16h11:11s |
| $C3$ | m | 01/01/2001 16h11:50s |
| $C3$ | w | 01/01/2001 16h12:40s |
| $C3$ | k | 01/01/2001 16h12:55s |

Figure 10: access log des sites $SF1$, $SF2$ et $SV$

2. Accepter la seconde page $p2$, demandée par $C$. Une fois cette étape terminée, on peut commencer à prédire le comportement de $C$.
3. Accepter la troisième page $p3$, demandée par $C$ et en fonction de sa demande précédente, ajouter sur $p3$ les liens vers les pages les plus demandées par ces prédecesseurs.

L'utilisation du Web Usage Mining Inter-Sites peut pourtant répondre à des impératifs plus exigeants que l'adéquation entre la catégorie à laquelle appartient l'utilisateur, et les liens qui lui sont proposés. Le fichier $Log+$ obtenu à partir des fichiers log de $SF1$ et $SV$, permet de tenir compte de l'historique de l'utilisateur observé, y compris sa navigation sur le site l'ayant envoyé sur $SV$. C'est en tenant compte de cet historique que l'on peut prétendre à fournir des liens dès l'arrivée du client sur le site $SV$, accordant ainsi un impact immédiat aux résultat obtenus après l'analyse du fichier $Log+$.

**L'impact immédiat des résultats**

Tenir compte de l'historique d'un client sur le site $A$ lorsqu'il demande une page sur le site $B$ permet des tentatives de prédictions dès son arrivée sur le site $B$. En effet, cet historique peut soit correspondre à un des comportements fréquents observés chez ses prédecesseurs, soit rester neutre. Dans le cas d'une correspondance, le site $B$ peut non seulement cibler le comportement du client en fonction de la catégorie à laquelle il appartient (cf paragraphe çi-dessus), mais surtout faire des propositions de pages à consulter pour cet utilisateur (ou modifier le site de la manière qui conviendra le mieux) en un nombre d'étape minimum. Ce paragraphe, présenté sous forme d'exemple, illustre les possibilités offertes par le Web Usage Mining Inter-Sites sur le plan des modifications dynamiques de liens hypertextes.

Reprenons les fichiers log illustrés par la figure 10. Les séquences de navigation des utilisateurs sur ces sites sont alors :

$C1$ : `<(a) (b) (c) (d) (To_SV)>` sur $SF1$ et `<(From_SF1) (i) (j) (l) (m) (w)>` sur $SV$
$C2$ : `<(a) (c) (d) (e) (To_SV)>` sur $SF1$ et `<(From_SF1) (i) (k) (l) (m) (w)>` sur $SV$
$C3$ : `<(f) (g) (h) (To_SV)>` sur $SF2$ et `<(From_SF2) (i) (z) (m) (w) (k)>` sur $SV$

Si l'on s'intéresse au partenariat existant entre les sites $SF1$ et $SV$, le fichier $Log+$ correspondant à ces deux sites et sur lequel nous allons procéder à une phase d'extraction de connaissances, est illustré de manière synthétique (i.e. sans les dates mais on conserve l'ordre d'apparition des objets demandés) par la figure 11.

| Client | d1 | d2 | d3 | d4 | d5 | d6 | d7 | d8 | d9 | d10 | d11 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $C1$ | a | b | c | d | To_SV | From_SF1 | i | j | l | m | w |
| $C2$ | a | c | d | e | To_SV | From_SF1 | i | k | l | m | w |

Figure 11: Fichier Log+, correspondant aux fichiers $LogSF1$ et $LogSV$ de la figure 10

Un processus de data mining executé avec un support $minSup$=100% sur ce fichier, permet d'obtenir le comportement fréquent suivant : `<(a) (c) (d) (To_SV) (From_SF1) (i) (l) (m) (w)>`. Les trois étapes nécessaires, décrites dans le paragraphe précedent, à la prise en compte des résultats avant d'exploiter les motifs séquentiels obtenus chez un client $C$ dont le comportement s'apparente au comportement fréquent de ses prédécesseurs, sont alors réduites à une seule étape : Accepter la demande faite par $C$ de la page $From\_SF1$ et renvoyer cette page modifiée (i.e. lui ajouter des liens vers les pages (i) et/ou (l) et/ou (m) et/ou (w)).
D'un point de vue global, les liens ajoutés à la première page demandée par $C$, seront dépendants du nombre et des types d'historiques fréquents que présentent les clients qui passent de $SF1$ a $SV$ comme l'illustre l'exemple 6.

**Exemple 6** *Considérons que deux fréquents sont issus de l'analyse du fichier Log+ entre $SF1$ et $SV$ :* `<(a) (b) (d) (To_SV) (From_SF1) (e) (g)>` *et* `<(a) (c) (d) (To_SV) (From_SF1) (f) (h)>`. *Si un client passe de $SF1$ à $SV$ (toujours en restant dans le cadre du partenariat) et présente à son arrivée sur $SV$ un historique qui correspond au premier fréquent, alors la première page qu'il consulte sur $SV$ ($From\_SF1$) peut être augmentée par des liens vers (e) et/ou (g). Son comportement sur $SF1$ a permi de le distinguer des utilisateurs qui ont eu un comportement correspondant au second fréquent (ou un comportement neutre).*

Ensuite le comportement de $C$ sur $SV$ permettra de prévoir son comportement à venir avec une précision supplémentaire par rapport au Web Usage Mining Intra-Site puisque l'historique sera encore pris en compte à chaque tentative de faire correspondre son comportement à celui de ses prédécesseurs.

## 4.2 Réduction

La première méthode que l'on peut envisager de mettre en place pour résoudre le problème du Web Usage Mining Inter-Sites, consiste à construire le fichier $Log+$ (fichier log du site virtuel $A \cap B$) selon l'algorithme 2. Une fois ce fichier obtenu, l'étape de data mining va alors analyser les données qu'il contient afin d'extraire les comportements fréquents des utilisateurs que le site $A$ a redirigé vers le site $B$. Ce paragraphe propose une approche plus performante, issue d'une analyse théorique mettant en jeu les propriétés des fréquents d'une part, et le support minimum d'un processus de data mining permettant de les obtenir sur le fichier log du site $A$, d'autre part.

### Adaptation de la problématique du Data Mining Incrémental

Les sites concernés par les applications du Web Usage Mining Inter-Sites analysent leurs fichiers access log, pour en extraire des comportements révélateurs, avec différents supports et de façon régulière, comme tout site le ferait dans le cadre du Web Usage Mining Intra-Site. Ce paragraphe présente

**fonction** *construireLog+*
Input: $Log_A$ et $Log_B$ les fichiers log des sites $A$ et $B$.
Output: $Log+$ le fichier log du site virtuel $A \cap B$.
**for** $C \in Log_A$ **do**
  **if** *dernierObjet(C)="To_ B"* **then**
    InscrireSequence($C$,$Log+$); //Inscrire la séquence de $C$ dans $Log+$
  **endif**
**endfor**
//Le fichier obtenu à la fin de cette première boucle sera désigné par $Log+_A$
**for** $C \in Log_B$ **do**
  **if** *premierObjet(C)="From_ A"* **then**
    InscrireSequence($C$,$Log+$); //Inscrire la séquence de $C$ dans $Log+$
  **endif**
**endfor**
//La partie ajoutée au fichier $Log+$ à la fin de cette seconde boucle sera désignée par $Log+_B$

Algorithm 2: Création du fichier $Log+$

une méthode destinée à exploiter les résultats obtenus de manière locale sur les sites mis en jeu, afin d'optimiser le processus de Web Usage Mining Inter-Sites.

La définition 7 présente la notion de $Log+_W$. Pour un site $W$ mis en jeu dans un processus de Web Usage Mining Inter-Sites, $Log+_W$ représente la partie du fichier $Log+$ qui correspond aux navigations des clients du site $W$ sur le site $W$. L'exemple 7 donne une illustration de ce concept.

**Définition 7** Soit $Log+$ le fichier log du site virtuel $A \cap B$ obtenu à partir des fichiers log des sites $A$ et $B$. $Log+_A$ représente la partie du fichier $Log+$ concernant la navigation des clients sur le site $A$ et $Log+_B$ la partie du fichier $Log+$ concernant la navigation des clients sur le site $B$.

**Exemple 7** *Si l'on considère à nouveau le fichier Log+ obtenu à la figure 11, alors le fichier* $Log+_{SF1}$ *qui lui correspond est illustré par la figure 12.*

| Client | d1 | d2 | d3 | d4 | d5 |
|---|---|---|---|---|---|
| $C1$ | a | b | c | d | To_SV |
| $C2$ | a | c | d | e | To_SV |

Figure 12: Fichier $Log+_{SF1}$

Pour la suite de ce paragraphe, $L^x_{DB}$ désigne l'ensemble des séquences fréquentes maximales sur DB avec un support minimum $x$ (i.e. $\forall s \in L^x_{DB}$, soit $n$ le nombre d'apparitions de $s$ sur $DB$, $n > x.|DB|$). De même, si une séquence $s$ a un nombre d'apparitions $n$ sur $|DB|$ alors son support est de $\frac{n}{|DB|}$.

**Propriété 1** $\forall x \in ]0..1]$, soit $s \in L^x_{Log+_A}$, $s$ se termine par $To\_B$.

La propriété 1 se vérifie de façon directe car 100% des séquences de $Log+_A$ se terminent par $To\_B$. A partir de cette observation, on peut déduire que tout fréquent extrait sur le fichier $Log+_A$ se termine obligatoirement par l'objet $To\_B$.
Le théorème 1 repose sur cette propriété et sur une manipulation des différents supports considérés lors de la phase d'extraction sur le fichier log de $A$ et dans le cadre du Web Usage Mining Inter-Sites.

**Théorème 1** Soient $x_1$ et $x_2$, deux supports tels que $x_2 = \frac{x_1.|Log+_A|}{|Log_A|}$ et soit $FreqLog+_A = \{s \in L^{x_2}_{Log_A}\}$ avec $s$ se terminant par $To\_B$, alors $FreqLog+_A = L^{x_1}_{Log+_A}$.

**preuve**

Prouvons que si $s$ est une séquence fréquente sur $Log+_A$ avec un support $x_1$ (i.e. $s \in L^{x_1}_{Log+_A}$), alors $s \in FreqLog+_A$ avec un support $x_2 = \frac{x_1.|Log+_A|}{|Log_A|}$.

Le nombre d'apparitions de $s$ sur $Log+_A$ est supérieur à $x_1.|Log +_A|$. Ce qui est également le nombre d'apparition minimum de $s$ sur $Log_A$ car $Log+_A \subset Log_A$. $s$ a donc sur $Log_A$ un support $x2 > \frac{x_1.|Log+_A|}{|Log_A|}$. De plus, d'après la propriété 1, $s$ se termine par $To\_B$. Donc $s \in FreqLog+_A$ car elle satisfait les règles de construction de $FreqLog+_A$. Donc $L^{x_1}_{Log+_A} \subseteq FreqLog+_A$.

Prouvons enfin que si $s \in FreqLog+_A$ avec un support $x_2 = \frac{x_1.|Log+_A|}{|Log_A|}$, alors $s$ est fréquente sur $Log+_A$ avec un support $x_1$ (i.e. $s \in L^{x_1}_{Log+_A}$) .

Si $s \in FreqLog+_A$ alors $s$ se termine par $To\_B$ (par construction de $FreqLog+_A$) et le nombre de séquences qui supportent $s$ dans $Log_A$ est le même dans $Log+_A$ (par construction de $Log+$). Soit $n$ le nombre d'apparitions de $s$ sur $Log+_A$, $n > x_2.|Log_A|$ donc $n > \frac{x_1.|Log+_A|}{|Log_A|}.|Log_A|$ et donc $n > x_1.|Log +_A|$. Donc $s$ est fréquente sur $Log+_A$ avec un support de $x1$ (i.e. $s \in L^{x_1}_{Log+_A}$). Donc $FreqLog+_A \subseteq L^{x_1}_{Log+_A}$.

Comme $L^{x_1}_{Log+_A} \subseteq FreqLog+_A$ et $FreqLog+_A \subseteq L^{x_1}_{Log+_A}$, alors $FreqLog+_A = L^{x_1}_{Log+_A}$ □

Le théorème 1 exprime le fait que si l'on envisage de lancer un processus d'extraction sur le fichier $Log+$ avec un support $x_1$, alors il existe un support $x_2$ pour lequel le fichier $Log_A$ recèle tous les fréquents du fichier $Log+_A$ calculés avec le support $x_1$. Pour retrouver ces fréquents, il suffit alors de prendre parmi les fréquents de $Log_A$ avec le support $x_2$, ceux qui se terminent par $To\_B$. Une fois ces fréquents obtenus (par une simple sélection parmi les fréquents déjà calculés par le site $A$) le processus de Web Usage Mining Inter-Sites dispose alors d'une information non négligeable, puisqu'il peut l'utiliser en tant que résultat d'un premier traitement sur le fichier $Log+_A$ et considérer $Log+_B$ comme un ajout à $Log+_A$ qui aboutit sur $Log+$. Ce raisonnement nous conduit alors vers la conclusion suivante : dans la mesure où l'on dispose déjà (grâce aux calculs fait par le site $A$, à une adaptation du support et à une sélection parmi les résultats) des résultats correspondants à un processus de data mining sur $Log+_A$ avec le support $x_1$, nous pouvons les utiliser comme données en entrée d'un processus de data mining incrémental pour calculer l'ensemble des fréquents de $Log+$ avec le support $x_1$.

Ce raisonnement est alors motivé (et sa mise en œuvre encouragée) par les temps de calcul améliorés que nous garantissent des méthodes incrémentales d'extraction de motifs séquentiels [15, 11].

### 4.3 Un algorithme pour le Web Usage Mining Inter-Sites : WUMIS

Dans ce paragraphe, après un rappel de la problématique de l'extraction incrémentale de motifs séquentiels, nous proposons une façon efficace de la résoudre via l'algorithme ISE+, une extension de l'algorithme ISE [11]. Nous présentons ensuite l'algorithme WUMIS qui est basé sur l'algorithme ISE+ et sur les résultats du théorème 1 pour résoudre le problème du Web Usage Mining Inter-Sites.

| Client | itemsets | | |
|---|---|---|---|
| $C1$ | 10 | 50 | 20 |
| $C2$ | 10 | 30 | 40 |
| $C3$ | 10 | 40 | 30 |
| $C4$ | 10 | 20 | |

$(DB)$

| Itemsets | |
|---|---|
| *70* | |
| *60* | *80* |
| *100* | *60* |
| *110* | *70* |

$(db)$

Figure 13: Une base de données $U$, constituée d'une base $DB$ et d'un incrément *db*

**Problématique de l'extraction incrémentale de motifs séquentiels**

Soit $DB$ une base de données et *minSupp* le support minimum. Soit *db* l'incrément dans lequel se trouvent les nouvelles transactions, ajoutées à $DB$ et les éventuels nouveaux clients. Soit $U = DB \cup db$, la base de données après la mise à jour (i.e. $U$ contient les séquences de $DB$ et *db*).
Soit $L_{DB}^{minSupp}$ l'ensemble des séquences fréquentes sur $DB$. Le problème de l'extraction incrémentale de motifs séquentiels consiste à déterminer $L_{U}^{minSupp}$, l'ensemble des séquences fréquentes sur $U$, en utilisant les connaissances apportées par $L_{DB}^{minSupp}$.

Pour illustrer cette problématique, considérons l'exemple de la figure 13. La base $DB$ exprime l'historique de 4 clients et *minSupp* est fixé à 50%. Les comportements fréquents alors obtenus sont les suivants : `<(10) (20)>`, `<(10) (30)>` et `<(10) (40)>`. Avec l'ajout de nouvelles séquences aux clients de $DB$, nous obtenons $U$. Les séquences fréquentes qu'il faut découvrir sur $U$ sont alors : `<(10) (20) (70)>`, `<(10) (30) (60)>` et `<(10) (40) (60)>`

Pour résoudre la problématique de l'extraction incrémentale de motifs séquentiels, nous avons proposé dans [11] un algorithme, appelé Ise, qui utilise les informations précédemment obtenues pour optimiser le processus d'extraction. Dans les grandes lignes, le comportement d'Ise peut être présenté de la façon suivante :
Soit $k$ la longueur des plus grandes séquences fréquentes dans $DB$, nous décomposons le problème de la recherche incrémentale de motifs séquentiels en deux étapes.

1. Rechercher toutes les nouvelles séquences fréquentes de taille $j \leq (k+1)$.
2. Utiliser les résultats issus de la première phase pour reprendre, à partir de $j$, un processus de type "Générer-Elaguer", jusqu'à trouver tous les fréquents sur $U$.

L'algorithme Ise+ que nous proposons, affiche des performances supérieures à celles d'Ise, grâce à une gestion plus fine des ensembles de fréquents mis en jeu et à une suppression de la seconde phase, évitant ainsi la reprise du processus "Générer-Elaguer".

**L'algorithme Ise+**

Pour résoudre la problématique de l'extraction incrémentale de motifs séquentiels, Ise+ manipule trois ensembles de séquences. Tout d'abord, les séquences de $DB$ qui peuvent devenir fréquentes si elles apparaissent dans *db*. Ensuite, les séquences qui n'apparaissent pas sur $DB$ mais qui sont fréquentes sur *db*. Enfin, les séquences fréquentes sur $U$ qui n'appartiennent à aucun des deux ensembles précédents (i.e. supportées par au moins une séquence de données, partagée entre $DB$ et *db*).

La figure 14 illustre la façon dont ces ensembles sont gérés. La première passe de Ise+ se situe sur *db* et permet d'identifier les items appartenant à *db*, ainsi que le nombre de leurs apparitions sur *db* et donc sur $U$ (par addition avec cette information sur $DB$). On peut alors, à l'issue de cette passe, déterminer les items fréquents sur $U$ qui apparaissent dans *db* : $L_1^{db}$. A partir de cet ensemble nous

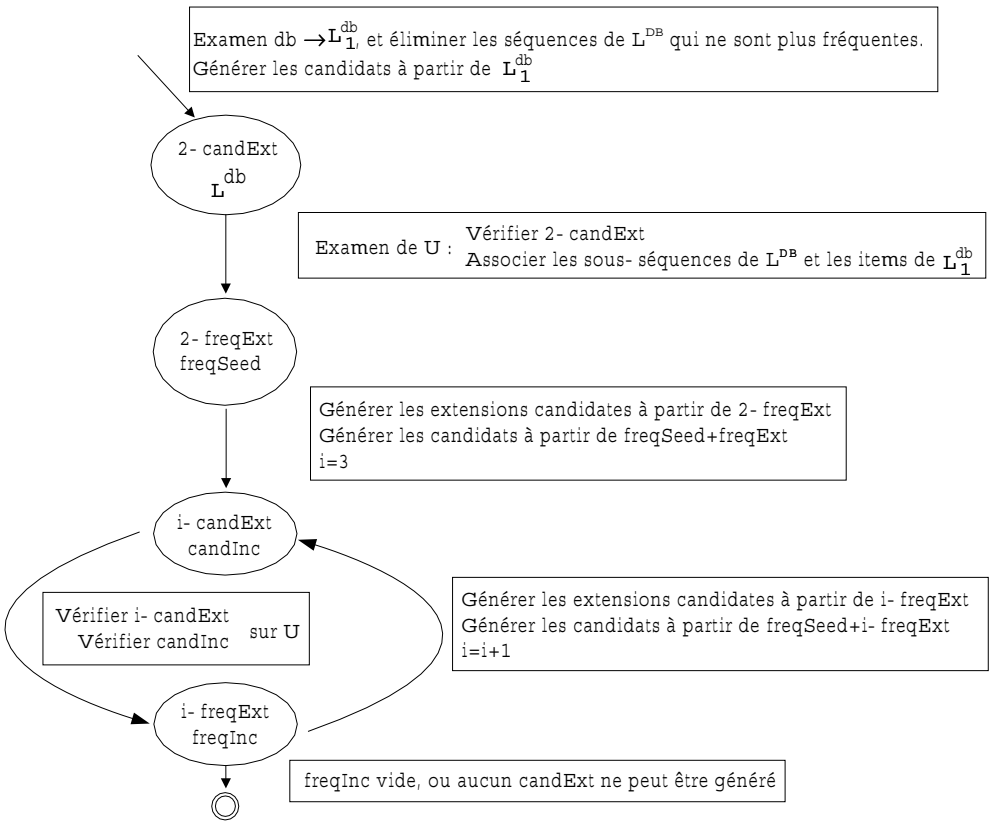


Figure 14: La gestion des trois ensembles de séquences par ISE+

construisons les séquences de taille 2 ($L_1^{db} \times L_1^{db}$), qui apparaissent sur $db$ (nouvelle passe sur $db$). Ces séquences forment l'ensemble 2-$candExt$.
Ces extensions sont alors vérifiées par une première passe sur $U$ pour donner l'ensemble 2-$freqExt$. ISE+ profite de cette passe pour associer aux sous-séquences de $L^{DB}$ les items fréquents de $L_1^{db}$, et vérifier la validité de ces associations (i.e. la sous-séquence $s$ de $L^{DB}$ est elle fréquente si on lui ajoute l'item $i$ de $L_1^{db}$ ?). Les associations validées sur $U$ permettent d'obtenir l'ensemble $freqSeed$ qui constitue la base des candidats qui seront désormais générés.

Les passes supplémentaires sont basées sur la méthode suivante : nous disposons de deux ensembles, 2-$freqExt$ et $freqSeed$. A partir de 2-$freqExt$ l'ensemble 3-$candExt$ est généré. A partir des ensembles 2-$freqExt$ et $freqSeed$, l'ensemble $candInc$ est généré selon le principe suivant : soit $s1 \in freqSeed$ et $s2 \in freqExt$, si le dernier item de $s1$ est égal au premier de $s2$, alors générer la séquence $s3$ en supprimant le premier item de $s2$ et en concaténant $s1$ et $s2$, enfin ajouter $s3$ à $candInc$. Il faut alors vérifier ces deux ensembles ($candExt$ et $candInc$) sur $U$.

Ce procédé est répété jusqu'à ce que l'on ne puisse plus générer d'extension candidate dans $candExt$ ou de séquences dans $candInc$. A la fin de ce processus, nous obtenons les séquences fréquentes sur $U$, à partir des séquences de $L^{DB}$ et les séquences maximales de $freqSeed \bigcup freqInc \bigcup freqExt$.

Le tableau 1 récapitule les notations utilisées dans l'algorithme ISE+.

Pour prouver qu'ISE+ construit l'ensemble des séquences fréquentes sur $U$, nous prouvons d'abord, dans les deux lemmes suivants, que tout nouveau fréquent peut être décomposé en deux sous-séquences.

| | |
|---|---|
| $L^{DB}$ | Séquences fréquentes sur $DB$. |
| $L_1^{db}$ | Séquences fréquentes de taille 1 sur $db$ et validées sur $U$. |
| $candExt$ | Séquences candidates générées à partir de $db$. |
| $freqExt$ | Séquences de $candExt$ validées sur $U$. |
| $freqSeed$ | Séquences fréquentes sur $U$, de la forme $<$(sous-séquence de $L^{DB}$) (séquence de $L_1^{db}$)$>$ . |
| $candInc$ | Séquences candidates générées à partir de $freqSeed$ et $freqExt$. |
| $freqInc$ | Séquences de $candInc$ validées sur$U$. |
| $L^U$ | Séquences fréquentes sur $U$. |

Table 1: Notations pour l'algorithme ISE+

La première est un fréquent de $DB$ et la seconde apparaît au moins une fois sur $db$.

**Lemme 1** *Soit $F$ une séquence fréquente sur $U$ telle que $F \notin L^{DB}$. Alors le dernier itemset de $F$ apparaît au moins une fois dans $db$.*

**Preuve:**

- si $|F| = 1$: comme $F \notin L^{DB}$, $F$ contient un itemset qui apparaît au moins une fois dans $db$, donc $F$ se termine avec un itemset qui apparaît au moins une fois dans $db$.
- si $|F| > 1$: soit S une séquence fréquente sur $U$, alors S$=< < A > < B > >$ avec $A$ et $B$ des séquences telles que $0 \leq |A| < |F|$, $0 < |B| \leq |F|$, $|A| + |B| = |F|$ et $B \notin db$. Soit $M_B$ l'ensemble des séquences qui contiennent $B$. Soit $M_{AB}$ l'ensemble des séquences qui contiennent $F$. Nous savons que si $|M_B| = n$ et $|M_{AB}| = m$ alors $\sigma \leq m \leq n$. De plus $M_{AB} \in DB$ (car $B \notin db$ et les transactions sont ordonnées dans le temps) donc $< < A > < B > >$ est fréquent sur $DB$ et $S \in L^{DB}$. Donc si une séquence fréquente $F$ sur $U$ ne se termine pas par un itemset de $db$, alors $F \in L^{DB}$ et, par opposition, si $F$ n'apparait pas dans $L^{DB}$, $F$ se termine par un itemset qui apparait au moins une fois dans $db$. □

**Lemme 2** *Soit $F = < < D > < S > >$, avec $D$ et $S$ deux séquences telles que $|D| \geq 0$, $|S| \geq 1$, une séquence fréquente sur $U$ telle que $F \notin L^{DB}$, alors $S$ apparaît au moins une fois dans db et $D$ est incluse dans (ou est) une séquence de $L^{DB}$.*

**Preuve:**

- si $|S| = |F|$: alors $|D| = 0$ et $D \in L^{DB}$.
- si $1 \leq |S| < |F|$: alors $D =< (i_1)(i_2)..(i_{j-1}) >$ et $S =< (i_j)..(i_t) >$ avec $S$ la sous-séquence maximale qui termine $F$ et qui apparaît au moins une fois dans $db$ (grâce au lemme 1 nous savons que $|S| \geq 1$). Soit $M$ l'ensemble de toutes les séquences qui contiennent $F$ et $time_u$ la date de mise à jour. $\forall s \in M$, $s =< < A > < B > >$ / $\forall i \in A$, $time_i < time_u$, $\forall j \in B$, $time_j \geq time_u$, nous avons $< (i_{j-1})(i_j)..(i_t) > \not\subset B$ (car $S$ est la sous-séquence maximale qui termine $F$ et qui apparaît au moins une fois dans $db$). Alors, comme $\forall i$, $M_i$ (la $i^{eme}$ séquence de $M$) contient $F$, et comme les transactions dans la base de données sont ordonnées dans le temps, $< (i_1)(i_2)..(i_{j-1}) > \subset A$. Donc $\forall s \in M$, $\exists A$ une sous-séquence qui commence $s$ telle que $A \in DB$, $D \subseteq A$. Donc $D \in L^{DB}$ □

Considérons les deux sous-séquences d'une nouvelle séquence fréquente. Nous prouvons grâce au lemme suivant, que la deuxième est générée en tant qu'extension candidate par ISE+.

**Lemme 3** *Soit $F$ une séquence fréquente sur $U$ telle que $F$ n'apparaît pas sur $L^{DB}$. $F$ être écrite $< < D > < S > >$ avec $D$ et $S$ deux séquences telles que $|D| \geq 0$, $|S| \geq 1$, $S$ apparaît au moins une fois dans db, $D$ est incluse dans (ou est) une séquence de $L^{DB}$ et $S \in candExt$.*

**Algorithm** ISE+
**Input:** $DB$ la base de données d'origine, $L^{DB}$ l'ensemble des séquences fréquentes sur $DB$, l'incrément $db$ et $minSupp$ le support minimum.
**Output:** $L^U$ l'ensemble des séquences fréquentes sur $U = DB \cup db$
**Method:**

*//première phase sur db*
$L_1^{db} \leftarrow \emptyset$
**foreach** $i \in db$ **do**
**if** $(support_{DB \bigcup db}(i) \geq minSupp)$ **then** $L_1^{db} \leftarrow L_1^{db} \bigcup \{i\}$;
**enddo**
2-$candExt \leftarrow L_1^{db} \times L_1^{db}$;
Eliminer de 2-$candExt$ les séquences $s/s \notin db$;
générer l'ensemble des sous-séquences de $L^{DB}$;
passe sur U : valider les 2-$candExt$ et construire $freqSeed$;
2-$freqExt \leftarrow$ frequent sequences from 2-candExt;

*// Phases suivantes*
j=2;
**While** (j-$freqExt$!=$\emptyset$) **do**
$candInc \leftarrow$ générer les candidats depuis $freqSeed$ et j-$freqExt$;
$j$++;
j-$candExt \leftarrow$ générer les candidats depuis j-$freqExt$;
Filtrer les séquences de j-$candExt$ $s/s \notin db$;
**if** (j-$candExt$!=$\emptyset$ OR $candInc$!=$\emptyset$) **then**
Valider j-$candExt$ et $candInc$ sur $U$;
**endif**
Mettre à jour $freqInc$ et j-$freqExt$;
**enddo**
$L^U \leftarrow L^{DB} \bigcup$\{séquences maximales de $freqSeed \bigcup freqInc \bigcup freqExt$\};
**end Algorithm** ISE+

Figure 15: L'algorithme ISE+

**Preuve:** Grâce au Lemme 2, nous devons seulement prouver que $S \in candExt$.

- si $S$ ne contient qu'une transaction, réduite à un item : $S$ est donc trouvée dans la première passe sur $db$ et ajoutée à 1-$candExt$.
- si $S$ contient plus d'un item : $candExt$ est construit suivant la méthode GSP à partir des extensions fréquentes apparaissant sur $db$. $candExt$ est donc un surensemble des séquences fréquentes sur $U$ qui apparaîssent sur $db$. □

Le théorème suivant garantit la validité de l'algorithme ISE+.

**Théorème 2** *Soit $F$ une séquence fréquente sur $U$ telle que $F \notin L^{DB}$. Alors $F$ est générée comme séquence candidate par* ISE+.

**Preuve:** A partir du Lemme 2 considérons les différentes déclinaisons possibles de $S$.

- si $S = F$: alors $S$ sera générée dans $candExt$ (Lemme 3) et ajoutée à $freqExt$.
- si $S \neq F$:

| | |
|---|---|
| \|D\| | Nombre de séquences (taille de la base) |
| \|C\| | Nombre moyen de transactions par client |
| \|T\| | Nombre moyen d'items par transaction |
| \|S\| | Longueur moyenne des fréquents de taille maximale |
| \|I\| | Longueur moyenne des itemsets dans les fréquents de taille maximale |
| $N_S$ | Nombre de fréquents de taille maximale |
| $N_I$ | Nombre d'itemsets fréquents de taille maximale |
| $N$ | Nombre d'items |
| $I^-$ | Nombre d'itemsets supprimés de $U$ pour construire $db$ |
| $D^{\%}$ | Pourcentage de transactions mises à jour dans $U$ |

Table 2: Paramètres

- si $S$ ne contient qu'une transaction, réduite à un item : alors l'association $<<D><(i)>>$ sera ajoutée dans $freqSeed$.
- si $S$ contient plus d'un item : considérons $i_{1_1}$ le premier item du premier itemset de $S$. $i_{1_1}$ est fréquent sur $db$, donc $<<D><(i_{1_1})>>$ sera généré dans $freqSeed$. D'après le Lemme 3, $S$ apparaît dans $freqExt$ et sera utilisée par ISE+ pour construire $<<D><S>>$ dans $candInc$ qui sera finalement ajouté à $freqInc$ □

**L'algorithme** WUMIS

Pour résoudre la problématique du Web Usage Mining Inter-Sites, nous proposons d'exploiter le théorème 1, grâce à l'algorithme WUMIS. Cet algorithme décrit dans la figure 4.3 procède en trois phases. Tout d'abord construire le fichier $Log+$. Ensuite calculer le support $x_2$ et construire l'ensemble des fréquents calculés sur $Log_A$ par le site $A$ avec ce même support et se terminant par $To\_B$. Enfin, cet ensemble étant prouvé identique à celui des séquences fréquentes sur $Log+_A$, WUMIS l'utilise comme paramètre pour l'algorithme ISE+, qu'il va alors faire fonctionner sur le fichier $Log+$, avec pour incrément $Log+_B$.

**Algorithm** WUMIS
**Input:** $Log_A$ et $Log_B$ les fichiers log des sites $A$ et $B$. $x_1$ le support minimum.
**Output:** $L^{x_1}_{Log+}$ l'ensemble des séquences fréquentes sur $Log+$
**Method:**

//Première phase : $Log+$
$(Log+, Log+_A, Log+_B) \leftarrow$ construireLog+($Log_A$, $Log_B$);
//Seconde phase sur $Log_A$
$x_2 \leftarrow \frac{x_1.|Log+_A|}{|Log_A|}$;
$L^{x_1}_{Log+_A} \leftarrow s/(s \in L^{x_2}_{Log_A}$ et $s$ se termine par $To\_B)$;
//Dernière phase sur $Log+$
$L^{x_1}_{Log+} \leftarrow$ ISE+ ($Log+_A$, $L^{x_1}_{Log+_A}$, $Log+_B$, $x_1$);

**end Algorithm** WUMIS

Figure 16: l'Algorithme WUMIS

| Name | \|C\| | \|I\| | N | \|D\| |
|---|---|---|---|---|
| C9-I4-N2K-D100K | 9 | 4 | 2,000 | 100,000 |
| C13-I3-N20K-D500K | 13 | 3 | 20,000 | 500,000 |
| C15-I4-N30K-D600K | 15 | 4 | 30,000 | 600,000 |
| C20-I4-N2K-D800K | 20 | 4 | 2,000 | 800,000 |

Table 3: Paramètres des jeux de données utilisés

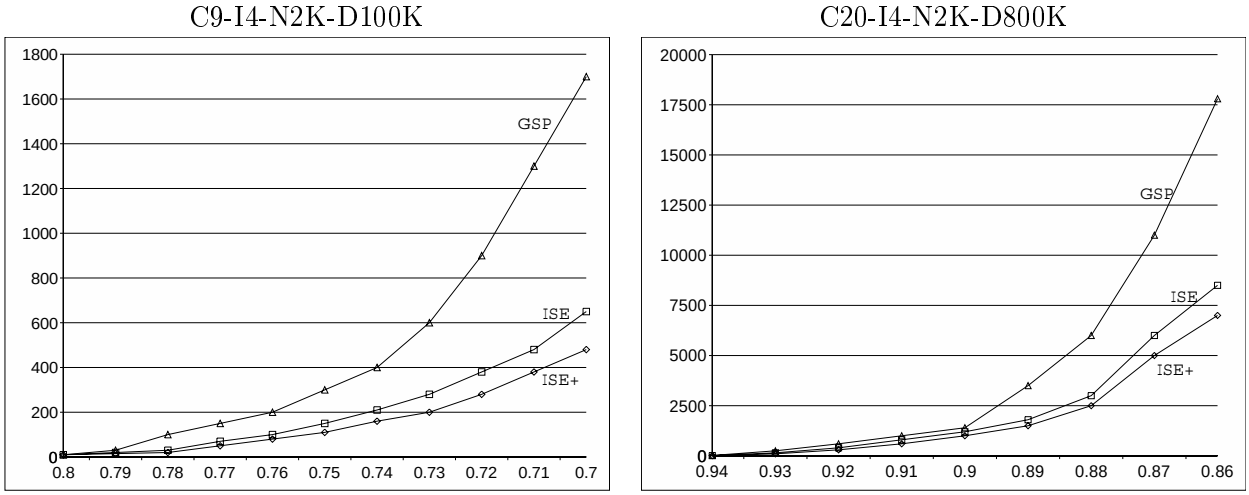


Figure 17: Temps d'exécution 100% des séquences mises à jour

**Expérimentations**

Les deux premières étapes de l'algorithme WUMIS étant linéaires, nous nous focalisons dans cette partie sur une comparaison entre l'algorithme GSP défini par [18] et l'algorithme ISE+. Les expériences ont été réalisées sur des jeux de données synthétiques dont les caractéristiques sont décrites dans les tableaux 2 et 3. Les données synthétiques ont été générées de la manière suivante. Des bases de données de taille $|U| = |DB + db|$ ont d'abord été générées et, de manière à examiner le comportement d'ISE+, des itemsets ont été supprimés de la base $U$ en utilisant les paramètres $D^{\%}$ et $I^{-}$ définis par l'utilisateur. De manière à comparer les deux algorithmes, nous examinons les temps d'exécution, en faisant varier le support minimal, de l'algorithme GSP sur la base $U$ et les temps d'exécution d'ISE+ en cumulant le temps d'application de GSP sur $DB$ et d'ISE+ sur $db$. La figure 17 illustre les résultats des expérimentations réalisées sur deux jeux de données : C9-I4-N2K-D100K et C20-I4-N2K-D800K, avec 100% des clients concernés par une mise à jour. Nous pouvons y observer l'efficacité de l'algorithme ISE+ qui peut fournir des résultats avec des temps d'exécution divisés par deux, en comparaison de GSP sur la base de données $U$.
Pour mettre en évidence les raisons du succès d'ISE+, nous avons réalisé une étude sur les fichiers C13-I3-N20K-D500K et C15-I4-N30K-D600K. Les résultats, exprimés à la figure 18, montrent non seulement des temps d'exécution plus performants pour l'algorithme ISE+, mais illustrent également une explication pour ces temps de réponse améliorés : le nombre de candidats générés par ISE+ en fin de processus est proportionellement inférieur au nombre de candidats générés par GSP. Ce que nous pouvons constater grâce à la figure 18, se situe donc au niveau du nombre de candidats générés par GSP (jusqu'à 7000 dans le premier cas et 14000 dans le second) et ISE+ (jusqu'à 4000 dans le premier cas et 8000 dans le second).

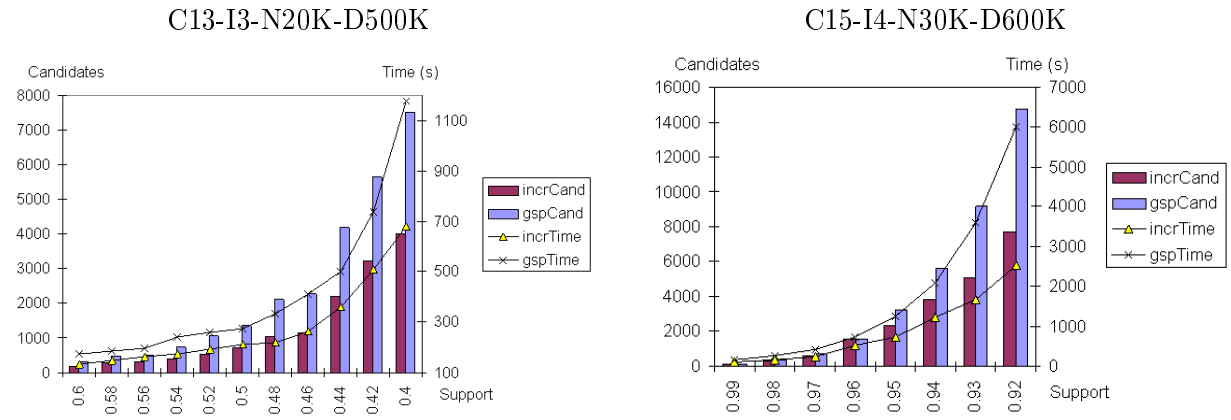


Figure 18: Nombre de candidats et temps d'exécution : mise en relation

## 5 Conclusion

Dans cet article, nous avons proposé une nouvelle problématique, le Web Usage Mining Inter-Sites, qui concerne l'analyse du comportement des utilisateurs de serveurs Web partenaires et nous avons détaillé l'importance de ses enjeux. Nous avons également proposé une solution efficace basée à la fois sur l'existence de corrélations entre les propriétés des connaissances extraites avec le support minimal et l'utilisation d'un algorithme de recherche de motifs séquentiels incrémental.
Les expériences réalisées sur des jeux de données synthétiques ont montré l'efficacité de l'approche.

Nous sommes actuellement en train d'étudier comment généraliser l'approche pour prendre en compte 3 puis n sites partenaires en évitant la solution naïve qui consiste à tester les sites deux à deux. D'un autre côté, nous nous intéressons à la recherche du comportement des utilisateurs notamment lors de retours en arrière fréquents. Même s'il existe actuellement des solutions basées sur des Applets Java, une topologie du site [16] ou des "client-site" log files [19], elles sont trop contraignantes et nécessitent que l'outil d'extraction puisse avoir accès aux données stockées sur le site du client.

## References

[1] 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.

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

[3] D.W. Cheung, J. Han, V.T. Ng, and C.Y. Wong. Maintenance of Discovered Association Rules in Large Databases: An Incremental Update Technique. In *Proceedings of the 12th International Conference on DataEngineering (ICDE'96)*, New-Orleans, Louisiana, March 1996.

[4] D.W. Cheung, S.D. Lee, and B. Kao. A General Incremental Technique for Maintening Discovered Association Rules. In *Proceedings of the Fifth International Conference on Database Systems for Advanced Applications (DASFA'97)*, Melbourne, Australia, April 1997.

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

[6] 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.

[7] R.W. Cooley. Web Usage Mining: Discovery and Application of Interesting Patterns from Web Data. Technical report, University of Minnesota, 2000.

[8] 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.

[9] M.Y Lin and S.Y Lee. Incremental Update on Sequential Patterns in Large Databases. In *Proceedings of the Tools for Artificial Intelligence Conference (TAI'98)*, pages 24–31, May 1998.

[10] F. Masseglia, P. Poncelet, and R. Cicchetti. An Efficient Algorithm for Web Usage Mining. *Networking and Information Systems Journal*, 2(5-6), December 1999.

[11] F. Masseglia, P. Poncelet, and M. Teisseire. Incremental Mining of Sequential Patterns in Large Databases. In *Actes des Journées Bases de Données Avancées (BDA'00)*, Blois, France, Octobre 1999.

[12] F. Masseglia, P. Poncelet, and M. Teisseire. Usinf Data Mining Techniques on Web Access Logs to Dynamically Improve Hypertext Structure. *ACM SigWeb Letters*, 8:13–19, October 1999.

[13] 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.

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

[15] S. Parthasarathy, M.J. Zaki, M. Ogihara, and S. Dwarkadas. Incremental and Interactive Sequence Mining. In *Proceedings of the 8th International Conference on Information and Knowledge Management (CIKM'99)*, pages 251–258, Kansas City, MO, USA, November 1999.

[16] 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.

[17] M. Spiliopoulou and L.C. Faulstich. WUM: A Tool for Web Utilization Analysis. In *Proceedings of the EDBT Workshop WebDB'98*, Valencia, Spain, March 1998.

[18] 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.

[19] 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.