HDM, un module de fouille de données distribué et temps réel

F. Masseglia[1,2] — **M. Teisseire**[1] — **P. Poncelet**[3]

[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*
[3] *Laboratoire LGI2P - École des Mines d'Alès - Site EERIE, Parc Scientifique Georges Besse 30035 Nîmes Cedex 1, France*
E-mail : {massegli,teisseire}@lirmm.fr, pascal.poncelet@ema.fr

RÉSUMÉ. *Les concepts relatifs à l'analyse du comportement des utilisateurs d'un site web ont pour objectifs la prévision des demandes, la réorganisation du site ou encore l'optimisation des performances réseaux. Les analyses évoquées sont basées sur une fouille a posteriori des informations collectées sur les actions que les utilisateurs suivent sur un site : le fichier log. Cependant, si l'on considère la vitesse à laquelle les comportements des utilisateurs peuvent varier, le résultat obtenu risque de perdre de sa pertinence s'il est exploité trop tard. Dans le but de contourner cette obsolescence nous proposons* HDM, *un module de fouille de données distribué, basé sur une heuristique et conçu pour obtenir les motifs de comportements fréquents pour répondre au problème du Web Usage Mining en temps réel.*

ABSTRACT. *The behaviour of a Web site's users may change so quickly that attempting to make predictions, according to the frequent patterns coming from the analysis of an access log file, becomes challenging. In order for the obsolescence of the behavioural patterns to become as null as possible, the ideal method would provide frequent patterns in real time, allowing the result to be available immediately. We propose a method to find frequent behavioural patterns in real time, whatever the number of connected users is.*

MOTS-CLÉS : *temps réel, fouille de données interactive, motifs séquentiels, approche heuristique et distribuée*

KEYWORDS: *real time, interactive data mining, zone mining, heuristic, distributed, long sequential patterns*

1. Introduction

Avec la popularité du World Wide 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'informations à partir de telles données est généralement appelée *Web Usage Mining*. Dans ce contexte, le problème consiste à rechercher des motifs comportementaux d'utilisateurs à partir d'un ou plusieurs serveur Web afin d'extraire des relations entre les données stockées. La plupart des techniques pour analyser ces motifs sont basées sur une fouille *a posteriori* des informations collectées, via les fichiers log, sur les actions que les utilisateurs suivent sur un site. Cependant, si l'on considère la vitesse à laquelle les comportements des utilisateurs peuvent varier, le résultat obtenu risque de perdre de sa pertinence s'il est exploité trop tard. Dans le but de contourner cette obsolescence nous proposons une nouvelle approche d'analyse appelée Web Usage Mining Temps Réel. L'objectif de cet article est de présenter HDM, un module de fouille de données distribué [MAS 01], basé sur une heuristique et conçu pour obtenir les motifs de comportements fréquents des utilisateurs en temps réel.

L'article est organisé de la manière suivante. Le paragraphe 2 propose un exposé de la problématique concernant la recherche de motifs comportementaux dans une base de données. Nous présentons également les limites des approches traditionnelles qui nous conduisent à proposer une nouvelle solution. Pour pouvoir répondre efficacement à la problématique, nous décrivons dans le paragraphe 3 notre proposition HDM. L'objectif du paragraphe 4 est de détailler l'architecture proposée. Enfin, dans le paragraphe 5, nous concluons en présentant les avantages de la proposition. Le lecteur intéressé par les expérimentations de l'approche HDM peut se reporter à [MAS 02] ou consulter `''http://www.kdd-tools.com''`.

2. Métaheuristique et fouille de données

Dans [AGR 95], le problème de la recherche de séquences dans de grandes bases de données est défini de la manière suivante : 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 (il s'agit de la représentation d'une transaction non datée). Une *séquence* est une liste ordonnée, non vide, d'itemsets (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. La problématique de la recherche de motifs séquentiels, appelé également motifs comportementaux (dans le contexte du Web Usage Mining, chaque item correspond à une URL accédé par un client), consiste à rechercher, en fonction d'un support minimum spécifié par l'utilisateur, les séquences de comportements dont le nombre d'occurrence est supérieur ou égal au support.

L'extraction de motifs comportementaux peut faire appel à deux types de solutions : Les algorithmes qui fournissent une solution complète, après avoir exploré l'espace des solutions possibles de la façon la plus pertinente possible. Les heuristiques qui approchent l'optimum global et affichent des temps d'exécution largement inférieurs en comparaison des algorithmes du premier point.

Les solutions du premier type sont nombreuses et basées soit sur des techniques comme la méthode générer-élaguer ([SRI 96, ZAK 98, MAS 99]), soit sur une exploration en mémoire de la base de données via des arbres suffixés comme dans [HAN 00] par exemple. Malheureusement, si l'espace des solutions augmente, de telle approches peuvent être mises en échec et ne fourniront "jamais" le résultat car l'exploration des possibilités devient trop coûteuse. Certaines méthodes proposent alors de fournir une solution très approchante (voire égale dans la plupart de cas) de l'optimum global [BAY 98, TOI 96, GUN 97]. Ces méthodes correspondent au deuxième type de solutions et ont pour but de répondre aux deux cas de difficultés majeures que l'on peut rencontrer : **La taille du résultat** interdit toute enumération des sous parties du résultat (ce que fait la méthode générer-élaguer). **La quantitié de données** (taille de la base) à explorer est bien trop grand pour employer une méthode de type "exploration en mémoire". Il existe également une autre raison qui limite l'utilisation des méthodes du premier type et qui est fortement liée à la problématique du Web Usage Mining : **une évolution rapide des données à traiter**. En effet, le site étudié peut voir le comportement de ses clients évoluer rapidement alors que son fichier access log est encore en cours d'analyse. Un algorithme de type incrémental (même appliqué aussi fréquement que son temps d'exécution le permet) ne peut prétendre être une solution temps réel. Tous ces éléments nous conduisent à proposer une nouvelle approche de résolution d'extraction de motifs comportementaux basée sur un algorithme stochastique d'optimisation combinatoire.

3. HDM (Heuristic based, Distributed Miner)

La figure 1 illustre une méthode possible pour résoudre le problème de l'extraction de motifs comportementaux dans les bases de données en utilisant une méta-heuristique. En considérant DB (étape 4) comme notre access log, il est effectivement possible d'employer une telle méthode pour découvrir les comportements fréquents des utilisateurs d'un site web. Le point de départ serait une population de fréquents réduite aux URL fréquentes, i.e. les plus demandées (étape 1). Ensuite des opérateurs de voisinage proposent un ensemble de solutions candidates (étapes 2 et 3) et leur validation sur la base de données (étape 4) permet de découvrir une solution plus fine (étapes 5 et 6). Si l'on applique à nouveau les opérateurs de voisinage sur ces fréquents, il est possible d'affiner encore les résultats et ce processus se répète jusqu'à stabilité de la solution proposée, i.e. l'ensemble des comportements fréquents.

Cette solution présente cependant un défaut majeur : le nombre de passes nécessaires sur la base de données pour évaluer chaque solution proposée est bien trop grand. En effet une méthode de type générer-élaguer ne demande que k passes, avec k représentant la taille du fréquent le plus grand, avant de fournir les résultats alors qu'une méta-heuristique demande plusieurs dizaines (voire centaines) de validations

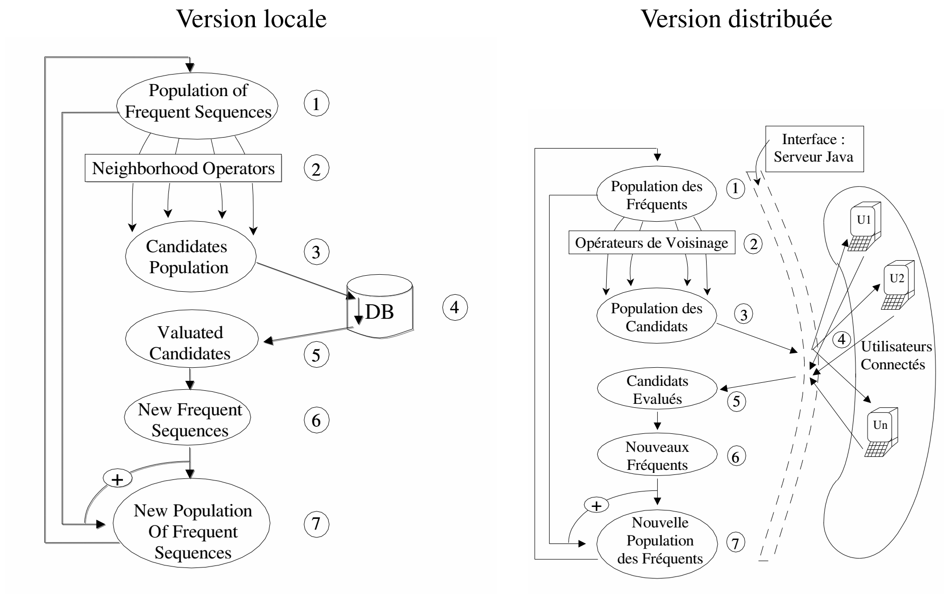

Figure 1. *Approche locale vs Approche distribuée*

sur la base de données (c'est la fonction objective, pilier des algorithmes stochastiques qui est censée s'exécuter en un temps minimum). Pour qu'une telle solution soit envisageable, il faudrait que la puissance de calcul disponible (pour comparer chaque candidat avec chaque séquence de la base) augmente en même temps que la taille de la base.

Pour contourner le problème lié à la puissance de calcul disponible, nous proposons de faire évaluer les candidats par chaque "acteur" de la base de données : les machines des utilisateurs connectés. En effet chaque utilisateur est actuellement en train d'utiliser un navigateur et donc une puissance de calcul qui lui est locale afin de naviguer sur ce site. Cette puissance de calcul est disponible si l'on offre l'architecture nécessaire pour l'exploiter. Ainsi la base de données représentée à l'étape 4 de la figure 1 (version locale) peut être remplacée par l'ensemble des machines connectées qui la composent (version distribuée).

Cette opération se fait en fournissant à la méta-heuristique une interface, qui simule une passe sur la base de données en contactant chaque machine connectée (Fig. 1). Chaque machine ainsi contactée va comparer sa propre séquence de navigation avec l'ensemble des candidats qu'elle vient de recevoir et renvoyer les résultats de cette comparaison. Cela revient à comparer chaque séquence de la base (qui regroupe les séquences des machines connectées) avec les séquences candidates, mais en un temps qui satisfait les conditions demandées par une fonction objective.

4. Architecture

Pour mettre en oeuvre la méthode présentée sur la page précédente, nous avons mis en place l'architecture décrite à la figure 2. Quand une machine demande l'accès à une page sur le serveur HTTP, celui-ci lui renvoie la page demandée avec une applet Java. Cette applet sera alors chargée de contacter le serveur Java, afin de recevoir les calculs à effectuer. Au début du cycle de vie du site, le premier client qui vient se connecter est le moteur de calcul, sur lequel se trouve l'algorithme stochastique. Le moteur va donc régulièrement envoyer au serveur des calculs à distribuer sur les machines connectées.

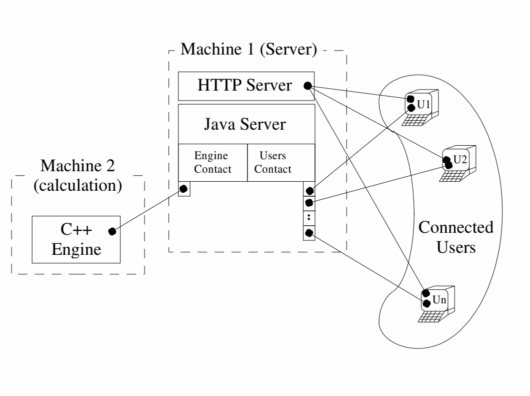

Figure 2. *Connexion entre le générateur de candidats et les clients via le serveur Web*

5. Opérateurs de voisinage et Evaluation des candidats

La principale idée, sur laquelle HDM se base, consiste à générer des populations de candidats grâce aux motifs fréquents, aux pages fréquentes et aux opérateurs de voisinage, puis d'envoyer ces candidats aux machines connectées afin d'évaluer leur pertinence (ou tout au moins leur distance par rapport à une séquence fréquente). Nous avons choisi des opérateurs de type génétique, aussi bien que des opérateurs basés sur les propriétés des motifs séquentiels fréquents. Quand nous faisons référence aux "séquences choisies aléatoirement" (ou bien "séquences aléatoires"), nous utilisons une roulette biaisée, telle que les séquences ayant le support le plus haut sont choisies plus fréquemment que les séquences de support plus faible.

Quand la génération de candidats est achevée, le moteur C++ envoie l'ensemble des candidats au serveur Java et ce dernier, au travers d'une technique de broadcast, le redistribue aux navigateurs connectés. Le calcul demandé aux navigateurs connectés est alors mis en œuvre par l'applet Java, lue au moment ou l'utilisateur se connecte sur le site, et consiste à comparer chaque séquence de l'ensemble des candidats reçus avec sa propre séquence de navigation. La comparaison consiste à obtenir un pourcentage, qui représente la distance entre le candidat et la séquence de navigation de l'utilisateur. Si la séquence candidate est incluse dans la séquence de navigation, le pourcentage sera de 100% et ce taux va décroitre avec l'apparition de parasites (différences entre le candidat et la séquence de navigation).

6. Conclusion

Au cours de cet article, nous avons proposé une nouvelle approche, appelée HDM dont la particularité est de permettre d'analyser en temps réel le comportement des utilisateurs d'un serveur Web. L'approche proposée est basée sur une approche de méta-heuristique. Pour pallier les problèmes liés à de telles approches, nous profitons de la puissance de calcul offerte par les utilisateurs connectés pour permettre de rechercher en temps réel quels sont les comportement fréquents parmi les utilisteurs. La méthode proposée offre les avantages suivants :

Une disponibilité immédiate : Etant calculés en temps réel, les résultats disponibles à un instant *t* reflètent, par nature, le comportement des utilisateurs connectés à ce même instant *t*.

Des résultats dédiés aux utilisateurs connectés : Etant donné que les résultats sont obtenus en temps réel, l'utilisateur final, i.e. le propriétaire du site concerné, peut utiliser les résultats fournis de manière immédiate. Cela permet de cibler et d'envisager les comportements des utilisateurs actuellement connectés, grâce à l'analyse du comportement global des utilisateurs connectés au même instant.

Une puissance de calcul inépuisable : Dans la mesure où HDM exploite la puissance de calcul disponible sur la machine de l'utilisateur connecté, le problème fournit lui-même la puissance nécessaire à le résoudre. Plus le problème s'étend plus il apporte de puissance.

Une fouille de données intéractive : A tout moment, le serveur Java peut accepter un nouveau support utilisateur et l'envoyer à HDM immédiatement. HDM prendra alors en considération les modifications avant d'analyser les résultats reccueillis chez les machines connectés et déterminer quelles séquences sont fréquentes. La prochaine population candidate sera alors proposée en fonction de ce nouveau support.

7. Bibliographie

[AGR 95] AGRAWAL R., SRIKANT R., « Mining Sequential Patterns », *Proceedings of the 11th International Conference on Data Engineering (ICDE'95)*, Tapei, Taiwan, March 1995.

[BAY 98] BAYARDO R., « Efficiently Mining Long Patterns from Databases », *Proceedings of the 1998 ACM SIGMOD Conference*, Almaden, USA, 1998, p. 85-93.

[GUN 97] GUNOPULOS D., MANILA H., SALUJA S., « Discovering all Most Specific Sentences by Randomized Algorithms Extended Abstract », rapport, 1997, Academy of Finland.

[HAN 00] HAN J., PEI J., YIN Y., « Mining Frequent Patterns without Candidate Generation », *Proceedings of International Conference on Management of Data (SIGMOD'2000)*, Dallas, Texas, 2000.

[MAS 99] MASSEGLIA F., PONCELET P., TEISSEIRE M., « Extraction efficace de motifs séquentiels : le prétraitement des données », *Actes des Journées Bases de Données Avancées (BDA'99)*, Bordeaux, France, Octobre 1999.

[MAS 01] MASSEGLIA F., TEISSEIRE M., PONCELET P., « Real Time Web Usage Mining : a Heuristic based Distributed Miner », *Web Information Systems Ingineering (WISE'01)*, Kyoto, Japan, December 2001.

[MAS 02] MASSEGLIA F., « *Algorithmes et applications pour l'extraction de motifs séquentiels dans le domaine de fouille de données : de l'incrémental au temps réel* », Thèse de doctorat, Spécialité Informatique, Université de Versailles - Saint Quentin, Janvier 2002.

[SRI 96] SRIKANT R., AGRAWAL R., « Mining Sequential Patterns : Generalizations and Performance Improvements », *Proceedings of the 5th International Conference on Extending Database Technology (EDBT'96)*, Avignon, France, September 1996, p. 3-17.

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

[ZAK 98] ZAKI M., « Scalable Data Mining for Rules », rapport n° PHD Dissertation, 1998, University of Rochester - New York.