Web Usage Mining : extraction de périodes denses à partir des logs

Florent Masseglia *, Pascal Poncelet**, Maguelonne Teisseire***, Alice Marascu *

* INRIA Sophia Antipolis, 2004 route des Lucioles - BP 93, 06902 Sophia Antipolis, France
{Alice.Marascu,Florent.Masseglia}@sophia.inria.fr
**EMA-LGI2P/Site EERIE, Parc Scientifique Georges Besse, 30035 Nîmes Cedex, France
{Pascal.Poncelet}@ema.fr
***LIRMM UMR CNRS 5506, 161 Rue Ada, 34392 Montpellier cedex 5 - France
{teisseire}@lirmm.fr

Résumé. Les techniques de Web Usage Mining existantes sont actuellement basées sur un découpage des données arbitraire (*e.g.* "un log par mois") ou guidé par des résultats supposés (*e.g.* "quels sont les comportements des clients pour la période des achats de Noël ? "). Ces approches souffrent des deux problèmes suivants. D'une part, elles dépendent de cette organisation arbitraire des données au cours du temps. D'autre part elles ne peuvent pas extraire automatiquement des "pics saisonniers" dans les données stockées. Nous proposons d'exploiter les données pour découvrir de manière automatique des périodes "denses" de comportements. Une période sera considérée comme "dense" si elle contient au moins un motif séquentiel fréquent pour l'ensemble des utilisateurs qui étaient connectés sur le site à cette période.

1 Introduction

L'analyse du comportement des utilisateurs d'un site Web, également connue sous le nom de Web Usage Mining, est un domaine de recherche qui consiste à adapter des techniques de fouille de données sur les enregistrements contenus dans les fichiers logs d'accès Web (ou fichiers "access log") afin d'en extraire des relations entre les différentes données stockées Cooley et al. (1999), Masseglia et al. (2003), Mobasher et al. (2002), Spiliopoulou et al. (1999). Ces derniers regroupent des informations sur l'adresse IP de la machine, l'URL demandée, la date, et d'autres renseignements concernant la navigation de l'utilisateur. Parmi les méthodes développées, celles qui consistent à extraire des motifs séquentiels Agrawal et Srikant (1995) s'adaptent particulièrement bien au cas des logs mais dépendent du découpage qui est fait des données. Ce découpage provient soit d'une décision arbitraire de produire un log tous les *x* jours (*e.g.* un log par mois), soit d'un désir de trouver des comportements particuliers (*e.g.* les comportements des internautes du 15 novembre au 23 décembre lors des achats de Noël). Pour comprendre l'enjeu de ces travaux, prenons l'exemple d'étudiants connectés lors d'une séance de TP. Imaginons que ces étudiants soient répartis en 2 groupes. Le groupe 1 était en TP le lundi 31 janvier. Le groupe 2 en revanche était en TP le mardi 1^{er} février. Chacun de ces

deux groupes de TP contient 20 étudiants et la nature du TP leur demande de naviguer selon un schéma précis. Selon le découpage des logs à raison d'un sous-log par mois, il n'y a eu pour le mois de janvier que 20 (au maximum) comportements qui respectent ce schéma, parmi les 200000 navigations présentes dans le log. La même observation est faite pour le mois de février. Ces navigations concentrées sur une période de deux jours ont alors trop peu de chance d'être découvertes en observant un log d'un mois ou d'un an. La méthode que nous proposons dans cet article permet de détecter qu'une (au moins) période dense existe et qu'elle prend place du 31 janvier jusqu'au 1^{er} février. De plus, cette période concerne 340 utilisateurs (connectés dans cet intervalle). Le comportement observé (la navigation demandée lors de la séance de TP) a donc un support de 11% dans cet intervalle (40 utilisateurs sur les 340 connectés).

2 Motifs séquentiels et Web Usage Mining

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 Agrawal et Srikant (1995), Masseglia et al. (1998).

Définition 1 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*. 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=$< (a)\ (d\ e)\ (h) >$, la séquence de données représentant les achats de ce client. S peut être interprétée par "C a acheté l'item a, puis en même temps les items d et e et enfin l'item h".*

Définition 2 Le $support$ de s, noté $supp(s)$, est le pourcentage de toutes les séquences dans une base DB 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*.

Les principes généraux de l'utilisation des motifs séquentiels dans le cas des logs d'accès Web sont similaires à ceux du processus d'extraction de connaissances exposé dans Fayad et al. (1996). Les données brutes sont collectées dans des fichiers access log des serveurs Web. La démarche se décompose en trois phases principales (prétraitement, fouille de données et interprétation par l'utilisateur des résultats obtenus).

3 Period : principe général

Dans cette section, nous présentons les motivations de notre travail, en termes d'intérêt des résultats visés et de difficultés engendrées, ainsi que le principe général de notre méthode.

Dans les grandes lignes, notre objectif est d'énumérer l'ensemble des périodes issues du log à analyser afin de déterminer quelles sont celles qui contiennent des motifs séquentiels fréquents. Soit C l'ensemble des clients du log et D l'ensemble des dates enregistrées.

Définition 3 *L'ensemble P des périodes possibles sur le log est défini de la manière suivante :* $P = \{(p_a, p_b)/(p_a, p_b) \in D \times D \; et \; a \leq b\}$.

Dans la définition suivante, nous considérons que $d_{min}(c)$ et $d_{max}(c)$ sont les dates d'entrée et de sortie de c dans le log (première et dernière action enregistrées pour c).

Définition 4 *Soit $C_{(a,b)}$ l'ensemble des clients connectés pendant la période (a, b). $C_{(a,b)}$ est défini de la manière suivante :*
$C_{(a,b)} = \{c/c \in C \; et \; [d_{min}(c)..d_{max}(c)] \cap [a..b] \neq \emptyset\}$.

Enfin, nous définissons les notions de *période stable* et de *période dense*. Dans le premier cas, il s'agit de chaque période maximale p_m sur laquelle C_{p_m} ne varie pas. Une *période dense* est une période stable qui contient au moins un motif séquentiel fréquent.

Définition 5 *Soit P_{stab} l'ensemble des périodes stables, P_{stab} est défini de la manière suivante :*
$P_{stab} = \{(m_a, m_b)/(m_a, m_b) \in P \; et$
1) $\nexists \; (m'_a, m'_b)/(b - a) < (b' - a') \; et \; [a'..b'] \cap [a..b] \neq \emptyset \; et \; C_{(m'_a,m'_b)} = C_{(m_a,m_b)}$
2) $\forall (x, y) \in [a..b], \forall (z, t) \in [a..b]/x \leq y, z \leq t \; on \; a \; C_{(x,y)} = C(z, t).\}$

Dans la définition 5, la condition 1 exprime le fait qu'il n'existe pas de période plus grande qui soit en "contact" avec P_{stab} et qui concerne les mêmes clients. La condition 2 exprime le fait que toutes les périodes de P_{stab} concernent les mêmes clients.

Définition 6 *Une période stable p est dite dense si C_p contient au moins un motif séquentiel fréquent respectant le support minimum spécifié par l'utilisateur proportionnellement à $|C_p|$.*

Extraire les motifs séquentiels fréquents sur chacune de ces périodes avec une méthode classique n'est pas une solution envisageable en raison du nombre de périodes stables (environ 2 millions pour 14 mois de log dans nos expérimentations). Dans la mesure où notre proposition est basée sur une heuristique, notre but est de fournir un résultat répondant aux critères suivants : Pour chaque période p appartenant à l'historique du log, soit $resultatReel$ le résultat à obtenir (le résultat qu'obtiendrait un algorithme de fouille de données qui explore tout l'ensemble des solutions après analyse des clients de C_p). $resultatReel$ est alors l'ensemble des motifs séquentiels à trouver. Soit $resultatPer$ les résultats obtenus par la méthode proposée dans cet article. Nous voulons minimiser $\sum_{i=0}^{taille(resultatPer)} S_i/S_i \notin resultatReel$ tout en maximisant $\sum_{i=0}^{taille(resultatReel)} R_i/R_i \subseteq resultatPer$. En d'autres termes, nous voulons trouver toutes les séquences apparaissant dans $resultatReel$ tout en évitant que le résultat soit plus grand qu'il ne le devrait (sinon l'ensemble de toutes les séquences, de toutes les navigations, pourrait constituer un résultat, car il englobe le résultat réel). Cette heuristique emprunte aux algorithmes génétiques leur conception du voisinage, en y intégrant les propriétés des motifs fréquents pour optimiser les candidats proposés. La principale idée, sur laquelle PERIO se base, consiste à parcourir l'ensemble P_{stab} des périodes stables et, pour chaque période p de

P_{stab}, à générer des populations de candidats grâce aux items fréquents et aux opérateurs de voisinage. Ensuite ces candidats sont comparés avec les séquences de C_p afin d'évaluer leur pertinence (ou tout au moins leur distance par rapport à une séquence fréquente). L'heuristique PERIO est décrite par l'algorithme suivant :

Algorithm *Perio*
In : P_{stab} l'ensemble des périodes stables.
Out : SP les motifs séquentiels correspondants aux comportements fréquents.
For ($p \in P_{stab}$)
  $itemsSupports$=getItemsSupports(C_p) ; // Maintenir le support des items
  $candidats$=voisinage(SP, $pagesSupport$) ; // Générer des candidats
  For ($c \in candidats$)
    For ($s \in C_p$) Noter(c, s) ;
  For ($c \in candidats$)
    If (support(c) $> minSupport$ OU $critere$) inserer(c, SP) ;
End algorithm *Perio*

Algorithm *Noter*
In : c le candidat à évaluer et s la séquence de navigation du client.
Out : $p[c]$ le pourcentage affecté à c.
  If ($c \subseteq s$) $p[c]$=100+taille(c) ;// c est incluse dans s, elle est favorisée
  If (taille(c)$\leq$ 2) $p[c]$=0 ;// c, de longueur 2, n'est pas incluse
  // Dans tous les autres cas, donner une note à c et défavoriser les distances accrues
  $p[c]$=$\frac{taille(PLSC(c,s))*100}{taille(c)} - taille(c)$;
End algorithm *Noter*

Pour chaque période de P_{stab}, PERIO génère les nouveaux candidats et compare ensuite chacun des candidats aux séquences de C_p. La comparaison consiste à obtenir un pourcentage qui représente la distance entre la séquence s du candidat et chaque séquence c de C_p. Si s est incluse dans c, le pourcentage sera de 100% et ce taux va décroitre avec l'apparition de "parasites" (différences entre le candidat et la séquence de navigation). Pour évaluer cette distance, le pourcentage est obtenu en divisant la taille de la plus longue séquence commune (PLSC) entre s et c par la taille de s. Par exemple, si s, de taille 4, contient une sous-séquence de taille 3 en commun avec c, alors la note de s pour c sera de $3/4$. De plus, dans le but d'obtenir des séquences les plus longues possible, nous utilisons un algorithme qui favorise les séquences les plus grandes, si elles sont incluses dans c. D'un autre côté, l'algorithme sanctionne les séquences trop longues si elles ne sont pas incluses (plus la séquences est longue, plus sa distance à la séquence de navigation sera pénalisante). Pour prendre en compte tous ces paramètres, le calcul effectué pour comparer les candidats à chaque séquence de C_p est décrit par l'algorithme NOTER. Enfin, les candidats évalués par l'algorithme NOTER seront insérés dans SP si leur support est supérieur au support minimum ou s'ils correspondent à un "critère de sélection naturelle". Ce critère, spécifié par l'utilisateur prend la forme d'un pourcentage qui définit la distance entre le support du candidat et le support minimum. Dans un nombre de cas (choisis de façon aléatoire), les candidats dont le support correspond au critère peuvent être insérés dans SP afin d'éviter à l'heuristique PERIOD de converger vers un optimum local.

4 Expérimentations

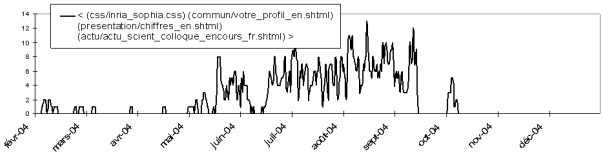

FIG. 1 – *Pics de fréquence pour un comportement sur une période longue*

Les programmes d'extraction sont réalisés en C++ sur des machines de type PC équipés de processus pentium 2,1 Ghz et exploités par un système RedHat. Nous avons effectué nos expérimentations sur les logs de l'Inria Sophia Antipolis. Ces logs sont découpés à raison de un log par jour. A la fin du mois, les logs journaliers sont regroupés sous forme d'un log mensuel. Nous avons donc travaillé sur les 14 logs mensuels disponibles, que nous avons considérés comme un seul log global recouvrant 14 mois d'enregistrements (de janvier 2004 à mars 2005). Ce log de 14 mois représente environ 14 Go de données. Il contient 3,5 millions de séquences (clients), la longueur moyenne de ces séquences est de $2,68$ et la longueur maximale est de 174 requêtes. Le log contient environ 2 millions de périodes et 300000 items. Le temps d'exécution de PERIOD sur ce log est d'environ 6 heures avec un support minimum de 2% (nous avons trouvé environ 400 groupes de comportements différents). Voici quelques exemples de comportements extraits :

$C1$ =<(semir/restaurant) (semir/restaurant/consult.php) (semir/restaurant/index.php) (semir/restaurant/index.php)>

$C2$ =<(eg06) (eg06/dureve_040702.pdf) (eg06/fer_040701.pdf) (eg06)>

$C3$ =<(requete.php3) (requete.php3) (requete.php3)>

$C4$ =<(Hello.java) (HelloClient.java) (HelloServer.java)>

$C5$ =<(mimosa/fp/Skribe) (mimosa/fp/Skribe/skribehp.css) (mimosa/fp/Skribe/index-5.html)>

$C6$ =<(sgp2004) (navbar.css) (submission.html)>

$C7$ =<(css/inria_sophia.css) (commun/votre_profil_en.shtml) (presentation/chiffres_en.shtml) (actu/actu_scient_colloque_ encours_fr.shtml)>

Le comportement $C1$ correspond à une navigation typiquement périodique. En effet, la cantine de l'Inria Sophia Antipolis a été en travaux pendant cette période et les membre de l'unité pouvaient, via les pages web "semir/restaurant", commander leurs repas froids de la semaine. Le comportement $C2$ correspond à une navigation sur des pages relatives aux "états généraux" de la recherche. $C3$ se trouve sur le site du projet "mascotte" et $C4$ sur celui de "oasis". Ces deux comportements correspondent à la consultation de pages de TD/TP sur les sites des chercheurs qui les proposent. Le comportement $C5$ est interprété par l'auteur des pages comme une conséquence de nombreux échanges en mars 2004 sur la mailing-list de Skribe. Avec le préfixe "geometrica/events/" pour $C6$. Le comportement $C6$ connaît deux pics (début avril et mi-avril). Il s'agit d'un comportement relatif à la soumission d'articles pour le symposium sgp2004, dont la date limite de soumission était le 7 avril pour les résumés et le 14 avril pour les articles.

Certains des comportements que nous avons mis en évidence n'ont pas un caractère ponctuel et se retrouvent sur plusieurs semaines, voir plusieurs mois. C'est le cas par exemple de $C7$ dont l'évolution est tracée dans la figure 1. On y observe que ce comportement se retrouve principalement sur 5 mois consécutifs (de mai à septembre).

5 Conclusion

Nous avons montré qu'avec une heuristique indexant le log période après période, nous pouvions extraire les comportements fréquents (quand il y en avait). Ces comportements ont la particularité d'être ponctuels ou répétitifs, en majorité rares et surtout représentatifs de périodes denses. Nos expérimentations ont permis d'extraire, entre autres, des comportements relatifs à des séances de TP ou encore des navigations sur les pages d'une conférence dans les jours qui précédent la date de soumission.

Références

Agrawal, R. et R. Srikant (1995). Mining Sequential Patterns. In *Proceedings of the 11th Int. Conf. on Data Engineering (ICDE'95)*, Tapei, Taiwan.

Cooley, R., B. Mobasher, et J. Srivastava (1999). Data preparation for mining world wide web browsing patterns. *Knowledge and Information Systems 1*(1), 5–32.

Fayad, U., G. Piatetsky-Shapiro, P. Smyth, et R. Uthurusamy (Eds.) (1996). *Advances in Knowledge Discovery and Data Mining*. Menlo Park, CA : AAAI Press.

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

Masseglia, F., D. Tanasa, et B. Trousse (2003). Diviser pour découvrir : une méthode d'analyse du comportement de tous les utilisateurs d'un site web. In *Les 19èmes Journées de Bases de Données Avancées*, Lyon, France.

Mobasher, B., H. Dai, T. Luo, et M. Nakagawa (2002). Discovery and evaluation of aggregate usage profiles for web personalization. *Data Mining and Knowledge Discovery 6*(1), 61–82.

Spiliopoulou, M., L. C. Faulstich, et K. Winkler (1999). A data miner analyzing the navigational behaviour of web users. In *Proceedings of the Workshop on Machine Learning in User Modelling of the ACAI'99 Int. Conf.*, Creta, Greece.

Summary

Existing Web Usage Mining techniques are currently based on an arbitrary division of the data. Those approaches have two main drawbacks. First, they depend on this arbitrary organization of the data. Second, they cannot automatically extract "seasons peaks" among the stored data. In this paper, we propose to perform a specific data mining process in order to automatically discover the densest periods.