Du e-commerce au m-commerce : vers une recommandation incrémentale

Armelle Brun et Anne Boyer

LORIA/Nancy Université - 615, rue du jardin botanique - 54506 Vandoeuvre les Nancy
{Armelle.Brun, Anne.Boyer}@loria.fr

RÉSUMÉ. *Les systèmes de recommandation, et notamment le filtrage collaboratif, sont traditionnellement utilisés dans les domaines du e-commerce et de la navigation web pour suggérer des ressources pertinentes aux utilisateurs au moment adéquat. Dans des approches dites "modèle", nous pouvons trouver les modèles à base d'usage et les règles d'association. Dans la littérature, ces modèles sont présentés comme des systèmes temps-réel. Ces dernières années, le domaine du m-commerce a émergé, dans lequel les recommandations sont diffusées sur un mobile au lieu de l'écran d'un ordinateur. Il faut donc adapter les techniques de recommandation aux nouvelles contraintes des terminaux mobiles. En particulier, puisque le respect de la vie privée est un objectif important, une façon de la préserver est de stocker les systèmes de recommandation sur le mobile. Cependant, bien que les systèmes de recommandation à base d'usage sont temps-réel, la génération des recommandations est complexe, et dans le cas où ils sont stockés sur le mobile, ils peuvent ne plus être temps-réel. Dans cet article, nous proposons un nouveau système de recommandation incrémental, à base d'usage, dans le but d'obtenir des recommandations instantanées dans le cadre du m-commerce.*

ABSTRACT. *Recommendation technologies, such as collaborative filtering, have traditionally been used in domains such as E-commerce and Web navigation to recommend resources to customers so as to help them to get the right resources at the right moment. In model-based approaches we can find the popular data mining models, as sequential association rules, that are usually presented as real-time recommenders. In the last few years, the m-commerce domain has emerged, that displays recommendations on the mobile device instead of the classical screen of the computer. In this paper user privacy preservation is an important objective and one way to be compliant with this constraint is to store the recommender on the mobile-side. However, although usage mining recommenders are real-time, many of them have a high complexity and when the recommender is implemented on a mobile device, it may not be real-time anymore. We put forward a new incremental recommender to get instantaneous recommendations when exploiting sequential association rules in the frame of m-commerce.*

MOTS-CLÉS : *Système de recommandation, complexité de calcul, recommandations incrémentales*

KEYWORDS: *Recommender systems, computation complexity, incremental recommendations*

1. Introduction

La démocratisation d'Internet et l'amélioration des technologies réseaux ont résulté en un accroissement du volume d'informations accessible à tout un chacun. Cependant, cette profusion d'informations résulte en un nombre toujours croissant d'utilisateurs insatisfaits. En effet, en raison de la trop grande quantité d'informations disponible, les utilisateurs ne peuvent plus accéder facilement à celle qu'ils recherchent. Un des objectifs des applications web actuelles est donc d'incorporer des mécanismes permettant aux utilisateurs un accès plus aisé aux informations qu'ils recherchent : celles qui correspondent à leurs attentes et à leurs goûts. L'intégration de ces mécanismes aura donc pour conséquence une augmentation de la satisfaction utilisateurs.
Dans des environnements de e-commerce (electronic commerce) ou de m-commerce (mobile commerce) par exemple, ces mécanismes sont un moyen de fidéliser les clients et par conséquent d'augmenter les ventes.

Les systèmes de recommandation sont un exemple de tel mécanisme, ils fournissent aux utilisateurs des recommandations personnalisées de produits ou de ressources en exploitant soit la connaissance que le système a des utilisateurs (Burke *et al.,* 1996), soit les actions passées de ces derniers, dans le but d'en déduire des informations sur leurs goûts ou attentes.
Un système de recommandation exploitant les actions passées des utilisateurs peut exploiter le contenu des ressources que les utilisateurs ont consultées, cette approche est dite *par contenu* (Pazzani *et al.,* 2007). Il peut également exploiter les traces d'usage des utilisateurs : quel utilisateur a consulté quelles ressources, cette approche est appelée *filtrage collaboratif* (Goldberg *et al.,* 1992).
L'approche par filtrage collaboratif (FC) ne requiert pas d'information *a priori* sur les ressources. Ainsi, des ressources de toute nature, de tout langage ou media (texte, audio, vidéo) peuvent être exploitées. Celle-ci ne requiert pas non plus d'information *a priori* sur les utilisateurs. La seule information utilisée ici est l'ensemble des ressources consultées par chaque utilisateur (les traces d'usage), complétée éventuellement par l'appréciation des utilisateurs sur ces ressources (les votes). Le succès du filtrage collaboratif ces dernières années est en partie dû à ces caractéristiques.

Sachant les informations sur les ressources et les utilisateurs, les algorithmes de filtrage collaboratif peuvent être classifiés en algorithmes *mémoire* et algorithmes *modèle*. Lorsqu'une recommandation est requise pour un utilisateur, que l'on appellera utilisateur actif, un système de recommandation *mémoire* (Sarwar *et al.,* 2000) évalue les similarités entre utilisateurs (cette opération est effectuée en-ligne) et ensuite recherche les voisins les plus proches de l'utilisateur actif pour calculer les recommandations. Le calcul des similarités et la génération de recommandations se font tous deux en-ligne. Etant donné le grand nombre de traces d'usage à traiter, cette approche souffre d'un problème de passage à l'échelle. Le temps de calcul requis est donc grand et les recommandations peuvent ne pas être disponibles au moment exact où elles sont demandées, la satisfaction des utilisateurs peut donc être impactée.

Un système de recommandation *modèle* (Breese *et al.,* 1998) crée un modèle des traces hors-ligne, ce qui permet des calculs plus conséquents. Lorsqu'une recommandation est requise pour l'utilisateur actif, le système de recommandation applique ce modèle pour générer des recommandations. La partie "coûteuse" de ce processus est donc exécutée hors-ligne. La partie en-ligne a une complexité en temps moindre et les recommandations sont calculées en un temps largement réduit.
Ces dernières années, l'intérêt porté aux algorithmes de data mining dans le contexte des systèmes de recommandation (Bozdogan, 2004) s'est agrandi, en raison de leur succès et leurs performances dans de nombreux domaines d'application. Ces algorithmes ont donc retenu notre attention et dans cet article nous nous intéressons aux systèmes de recommandation *modèle* exploitant les algorithmes de data mining, et plus particulièrement les règles d'association.

Bien que les approches modèle pallient partiellement les problèmes de passage à l'échelle des systèmes de recommandation mémoire, la question du temps de calcul des recommandations reste toujours d'actualité : peuvent-elles réellement être générées en un temps restreint ? Les utilisateurs peuvent avoir besoin de recommandations à tout moment, le système de recommandation doit donc leur fournir des recommandations au moment où elles sont requises.
Les systèmes de recommandation modèle et tout particulièrement ceux à base de data mining sont souvent présentés comme des systèmes temps-réel dans la littérature (Anand *et al.,* 2004, Huang *et al.,* 2006). La propriété temps-réel peut cependant dépendre du nombre de ressources manipulées ainsi que de la capacité de calcul de l'ordinateur sur lequel le système de recommandation est implanté. Qu'est-ce qu'un système temps-réel ? Un système temps-réel "met à jour les informations à la même vitesse qu'il reçoit les données" [1]. Dans le cadre de systèmes de recommandation à base de filtrage collaboratif, les données reçues par le système sont l'ensemble des consultations utilisateurs. De nouvelles données sont donc disponibles à chaque fois qu'un utilisateur consulte une ressource. Ainsi, un système de recommandation temps-réel doit générer des recommandations entre deux consultations d'un même utilisateur.

Pour des raisons de commodité et de sécurité, les systèmes de recommandation sont habituellement implantés sur le serveur qui a une grande capacité de calcul. Dans ce cadre, sachant la dernière ressource consultée par l'utilisateur actif, les systèmes de recommandation peuvent généralement générer des recommandations avant que l'utilisateur ne consulte la ressource suivante. Ces modèles peuvent par conséquent être effectivement considérés comme temps-réel et bien adaptés à des applications Web, e-commerce ou m-commerce. Supposons cependant que des recommandations soient générées très peu de temps avant que l'utilisateur consulte sa ressource suivante, ce système sera considéré comme temps-réel, mais les recommandations sont présentées trop tard à l'utilisateur.
Dans certains cas, un système de recommandation peut être implanté du coté du client. Dans le cadre du m-commerce par exemple, le système peut se trouver sur le mobile. Qu'en est-il dans ce cas du temps de calcul des recommandations ?

1. traduction de http ://www.thefreedictionary.com

Le cadre d'application de cet article est le m-commerce, et plus particulièrement dans un contexte de supermarché. Dans ce cadre, nous avons choisi d'implanter le système de recommandation sur le mobile pour des raisons de respect de la vie privée. En raison de la faible capacité de calcul des mobiles, nous nous posons donc la question de savoir si les systèmes de recommandation (plus particulièrement ceux exploitant des algorithmes de data mining) permettent de générer des recommandations en temps-réel lorsqu'ils sont implantés sur un mobile.
Pour répondre à cette question, nous nous intéressons tout particulièrement à la façon exacte dont l'approche par règle d'association calcule les recommandations. Nous proposons ensuite un nouveau système de recommandation incrémental qui permet de générer des recommandations en un temps plus restreint sans décroître la qualité des recommandations.

La section 2 se focalise sur les systèmes de recommandation en m-commerce. La section suivante décrit brièvement notre scénario d'application, le choix d'implantation que nous avons fait ainsi que les données manipulées par le modèle. La section 4 effectuera une présentation des systèmes de recommandation à base de règles d'association, puis explicitera en détails la façon dont les recommandations sont calculées dans le cas de l'exploitation de règles d'associations et montrera que ce calcul est une tâche complexe. Par la suite, cette section présentera un nouveau système de recommandation incrémental qui permet de réduire le temps de calcul des recommandations tout en conservant la même qualité de recommandations. Enfin, nous conclurons et présenterons quelques perspectives.

2. Les systèmes de recommandation en m-commerce

Le m-commerce est un domaine d'application récent qui a émergé dans la dernière décennie en raison de l'amélioration de la capacité des terminaux mobiles, de l'accroissement de la bande passante, des réseaux sans-fils ainsi que des réseaux mobiles. Le m-commerce peut être défini comme suit : "les applications en m-commerce ne couvrent pas seulement les applications du e-commerce, mais aussi les nouvelles applications qui peuvent être exécutées à tout moment, de n'importe quel endroit via des mobiles" (traduit de (Hu *et al.,* 2006)). Le m-commerce est actuellement considéré comme un complément et un remplaçant potentiel du e-commerce. La recherche dans le domaine des systèmes de recommandation en m-commerce s'est d'ailleurs accélérée ces dernières années (Tewari *et al.,* 2003, Tarasewich, 2003). Dans ce cadre, les applications sont nombreuses et variées, nous pouvons mentionner par exemple le tourisme (Wan, 2009) ou la recommandation dans le domaine de la restauration (Hosseini-Pozveh *et al.,* 2009).

Le m-commerce a cependant certaines différences avec le e-commerce, comme la mobilité, la capacité de calcul limitée, les capacités de transmission, la taille de l'écran, la durée de la batterie, etc. (Zenebe *et al.,* 2005).

Tout comme en e-commerce, en m-commerce le système de recommandation peut être implanté soit sur le serveur soit sur le client (mobile).
Dans (Wan, 2009), le système de recommandation est implanté sur le serveur, dans le contexte du tourisme. Le mobile enregistre les informations à propos des utilisateurs et les transmet au serveur qui se charge de calculer les recommandations. Celles-ci sont ensuite envoyées au mobile qui les affiche. Dans ce cadre, les problèmes rencontrés sont la sécurité (notamment la stabilité de la connexion) et la vie privée.
(Tveit, 2001) propose un système P2P pour pallier le problème du passage à l'échelle des systèmes de recommandation. Cette approche souffre du nombre de communications entre mobiles et de la capacité de communication des mobiles. De plus l'auteur met en avant le problème de la capacité de traitement des mobiles. Cette approche est également sensible aux attaques malicieuses.
Les systèmes de recommandation peuvent également être implantés côté client (Lee, 2004). Ces derniers souffrent des capacités de calcul mémoire du mobile. Cependant, la capacité de la bande passante, la sécurité et le respect de la vie privée ne sont plus un problème puisqu'il n'y a pas de communication entre le serveur et le mobile.
Nous pouvons noter ici que la taille de l'écran est un problème récurrent, quelque soit l'approche choisie.

3. Cas d'application

3.1. *Scénario et choix d'implantation*

Dans le cadre du e-commerce, les systèmes de recommandation observent le comportement des utilisateurs ainsi que leurs intérêts en exploitant leurs traces de navigation et d'achat, et leur recommandent les ressources pertinentes.

Ce papier se concentre sur les systèmes de recommandation en m-commerce dans un contexte de supermarché. Dans ce cas, le système de recommandation observe le comportement des utilisateurs (les produits qui les ont intéressés) et leur suggère des produits pertinents (liés à leurs intérêts passés, aux offres spéciales, etc.). Rappelons que les systèmes de recommandation auxquels nous nous intéressons ne requièrent aucune information *a priori* sur les ressources ou les utilisateurs (à l'exception des usages).
Pour effectuer leurs achats, les clients utilisent un caddie dans lequel ils déposent les produits qu'ils souhaitent acheter. Une puce RFID est placée sur chaque produit. Chaque caddie est équipé d'un mobile, fourni par le magasin. Le mobile sera le lecteur de puces RFID. Les clients ne sont donc pas obligés d'utiliser leur propre mobile, d'ailleurs ils peuvent ne pas en posséder. Lorsqu'un utilisateur dépose un produit dans son caddie ou qu'il en met tout simplement un dans sa main, le produit sera identifié par le lecteur. Cette information sera exploitée par le système qui générera des recommandations, et qui seront affichées sur l'écran du mobile.

Du point de vue du client, aucune contrainte n'est ajoutée, celui-ci effectue ses achats comme à son habitude. Tout le long de son trajet, le système de recommanda-

tion lui propose des produits qu'il juge être intéressants pour lui. Les recommandations qui lui sont faites à un moment donné sont calculées à partir des produits par lesquels il a été intéressé (dans sa session courante). L'utilisateur choisit un produit qui peut être ou non dans la liste de recommandation, le système lui présente ensuite une nouvelle liste de produits recommandés et ainsi de suite.

Lorsque l'utilisateur prend en main un produit, le système doit lui suggérer une nouvelle liste de recommandations en lien avec ses intérêts. Si le temps de calcul des recommandations est grand, il est possible qu'aucune recommandation ne soit disponible avant le moment où l'utilisateur prend un autre produit dans sa main. Si l'utilisateur souhaite une recommandation, il va devoir attendre, et sa satisfaction peut décroître. Les recommandations doivent donc être générées instantanément.

Dans ce contexte, nos priorités donc sont bien évidemment le temps de calcul, la qualité des recommandations mais aussi la préservation de la vie privée. Si le système de recommandation est implanté côté serveur, le grand nombre de communications entre le mobile et le serveur (dans les deux sens), associé à leur faible qualité et sécurité représentent un inconvénient. Dans ce cas la vie privée des utilisateurs peut ne pas être respectée. Implanter le système de recommandation du coté du client a l'avantage de préserver la vie privée des utilisateurs, car la recommandation et les actions des utilisateurs ne sont pas connues du serveur. C'est pour cette raison que dans cet article nous choisissons d'implanter le système de recommandation sur le mobile. Le problème subsistant dans ce cas est donc la capacité de calcul du mobile, problème auquel nous nous intéressons dans la suite de cet article.

3.2. *Les données et le modèle*

A chaque fois qu'un utilisateur s'intéresse à un produit : soit qu'il le dépose dans son caddie, soit qu'il le prend en main (et le repose dans le rayon ensuite), cette information est capturée par le mobile et est exploitée par le système de recommandation pour recommander des produits en lien avec ceux qui l'ont intéressé. Les clients n'ont pas besoin d'être authentifiés, et aucune information explicite les concernant ne leur est demandée, la seule information exploitée est la suite des produits par lesquels le client a été intéressé. La vie privée de l'utilisateur est donc préservée. De plus, l'information automatiquement collectée par le mobile est uniquement connue de lui, cette information n'est ni transmise à un autre mobile ni au serveur. Cette caractéristique représente un avantage puisqu'un utilisateur ne souhaite évidemment par voir l'information de son intérêt pour un produit qu'il n'a finalement pas acheté stockée sur un serveur.

Lorsque l'utilisateur arrive à la caisse, une partie de l'information contenue dans le mobile est transmise au serveur. Seule la liste des produits effectivement achetés par le client est transmise. Ceux par lesquels il a été seulement intéressé ne sont pas transmis. Dans le cas où l'utilisateur ne possède pas de carte de fidélité, l'information transmise est anonyme. Ainsi, notre approche n'aggrave pas la situation actuelle dans

les supermarchés concernant les informations transmises au serveur.
A l'opposé des approches traditionnelles dans le contexte des supermarchés, l'information enregistrée par le serveur n'est pas l'ensemble des produits achetés, mais la séquence des achats (l'ordre dans lequel les produits ont été déposés dans le caddie est connu).

Les données d'apprentissage sont donc composées de séquences de produits achetés par les utilisateurs. Le système de recommandation est ensuite construit sur le serveur. L'ordre d'achat est important afin de recommander à l'utilisateur des produits devant lesquels il va passer et non ceux qu'il a déjà vus. Le modèle peut être mis à jour périodiquement, chaque jour ou semaine, ou en fonction des promotions ou encore des modifications dans la disposition des rayons. Le modèle mis à jour intègre ainsi l'évolution des comportements utilisateur.

Etant donné qu'aucune identification n'est nécessaire, aucune donnée privée de l'utilisateur est stockée, et les utilisateurs sont plus confiants et plus à même d'exploiter le système. De plus, le modèle est uniquement stocké sur des mobiles possédés par le supermarché, il ne peut être utilisé par d'autres personnes ou magasins et les utilisateurs n'ont pas à installer de logiciel sur leur propre mobile. De plus, les communications avec le serveur sont quasi inexistantes, le système est donc plus sécurisé.

Nous pouvons noter que, étant donné que le système de recommandation est stocké sur le mobile, la mémoire nécessaire pour stocker le modèle peut ne pas être suffisante. Ce constat n'est pas un inconvénient puisque les mobiles sont fournis par le supermarché qui peut choisir des mobiles adéquats. Cela aurait pu cependant être un problème si l'on avait choisi d'installer le système de recommandation sur les mobiles personnels des clients.

Dans cette section nous avons répondu au problème du respect de la vie privée. Nous nous intéressons dans la section suivante au problème du temps de calcul des recommandations. Nous nous penchons sur ce problème en présentant dans un premier temps les règles d'association, approche pour laquelle nous présenterons en détails les différents calculs à effectuer pour générer des recommandations. Enfin, nous présenterons notre proposition de recommandations incrémentales pour diminuer le coût de ces calculs.

4. Vers un système de recommandation incrémental

4.1. *Les approches data mining*

De nombreuses approches ont été exploitées dans les systèmes de recommandation (Breese *et al.,* 1998, Adomavicius *et al.,* 2005), dont les approches data mining.

Le data mining consiste à "découvrir, analyser et extraire de la connaissance à partir d'une grande quantité de données" (Han *et al.,* 2001). Dans le cadre des systèmes de recommandation en e-commerce ou m-commerce, les données sont les traces d'usage (de navigation) et les algorithmes de data mining sont exploités pour modéliser le

comportement et les préférences des utilisateurs. Ces informations sont ensuite utilisées pour prédire les préférences et le comportement futur de ces utilisateurs.

Les approches data mining ont l'avantage de pouvoir être utilisées sans requérir l'authentification des utilisateurs, elles peuvent ainsi fournir des recommandations à des utilisateurs inconnus, ce qui est une situation courante dans la pratique. En effet, en e-commerce, m-commerce ou dans de nombreuses applications de navigation, il est très important de fournir des recommandations aux utilisateurs lorsqu'ils ne sont pas identifiés, non seulement pour leur faire des recommandations dès qu'ils arrivent sur le site (pour améliorer leur satisfaction) mais également pour attirer des clients potentiels. C'est également le cas dans le contexte de supermarché auquel nous nous intéressons, où il est important de pouvoir effectuer des recommandations à tous les utilisateurs, qu'ils soient connus du magasin ou non.

4.2. *Les règles d'association séquentielles*

Les règles d'association (RA), sont une des approches du data mining, elles ont été introduites par Agrawal et al (Agrawal *et al.,* 1993) dans le contexte de l'analyse du panier de la ménagère. Les règles d'association permettent de capturer des relations entre ressources dans les transactions d'achat.
Les règles d'association séquentielles (RAS) sont une évolution des règles d'association, dans lesquelles l'ordre de consultation des ressources est pris en compte. Les RAS ont été proposées par Srikant et al (Srikant *et al.,* 1996) pour extraire des relations ordonnées entre ressources. Les RAS sont également appelées motifs séquentiels et sont classiquement exploitées dans des applications de e-commerce et de m-commerce.

Une RAS est une expression de la forme $X \Rightarrow Y$, où X (appelé l'antécédent) et Y (le conséquent) sont des séquences de ressources. Habituellement $X \Rightarrow Y$ est considéré comme une RAS si son support et sa confiance sont supérieurs à deux seuils fixés *a priori*.

Dans le domaine des systèmes de recommandation, les RAS sont exploitées pour capturer les dépendances ordonnées entre ressources dans les sessions utilisateur. Une RAS signifie que lorsque les utilisateurs ont consulté la séquence de toutes les ressources de X, alors ils consultent généralement Y. Dans le cadre des systèmes de recommandation, étant donné que le but est de prédire la ressource suivante à consulter, Y n'est composé que d'une seule ressource.

Lors de l'apprentissage du modèle, qui se fait hors-ligne, les traces de navigation sont pré-traitées de façon à obtenir des sessions de consultation de ressources ordonnées. Ensuite, le système découvre et extrait les RAS, toujours hors-ligne, en fonction de leurs support et confiance. Le modèle est donc composé d'un ensemble de RAS. La plupart des algorithmes de découverte des RAS sont incrémentaux, comme GSP (Srikant *et al.,* 1996) ou les algorithmes inspirés de l'algorithme APriori, pour lesquels les RAS de taille $k+1$ sont déduites des RAS de taille k.

La génération des recommandations est quant à elle faite en-ligne. Le système de recommandation exploite la session de navigation de l'utilisateur actif ainsi que les RAS du modèle, qu'il essaie de mettre en correspondance pour évaluer le score de chaque ressource possible $r_m \in R$. Les ressources ayant obtenu les scores les plus élevés sont ensuite recommandées à l'utilisateur. Le processus de recommandation est donc totalement exécuté en-ligne.

La plupart des travaux traitent de RAS non contiguës, cela signifie qu'à l'apprentissage, les règles d'association apprises peuvent être composées de ressources non contiguës dans les sessions d'apprentissage. Au test les ressources considérées dans la session active peuvent ne pas être contiguës.

Nous pouvons noter ici, que de nombreux travaux se sont intéressés à la réduction de l'espace mémoire exploité pour stocker les RAS et améliorer le temps d'accès à celles-ci (Xiao *et al.,* 2001, Mobasher *et al.,* 2002, Nakagawa *et al.,* 2003). Ces travaux proposent de stocker les RAS sous la forme d'un arbre, réduisant ainsi le temps d'accès aux RAS. Cependant, la complexité de l'étape de génération des recommandations reste identique.

4.3. *La génération des recommandations*

Nous nous focalisons maintenant sur la façon exacte dont les recommandations sont calculées dans un système de recommandation exploitant des règles d'association. Nous montrons que ce calcul est complexe et que les recommandations peuvent être générées en un temps non négligeable. Nous proposerons ensuite une nouvelle façon de calculer les recommandations, en rendant le calcul de la recommandation incrémental.

A notre connaissance, aucun travail n'a été présenté sur la façon exacte dont la session de l'utilisateur actif est mise en correspondance avec le modèle. Nous présentons donc ici cette mise en correspondance et montrons que cette étape est complexe et peut prendre du temps.

Avant de présenter en détails la façon dont les recommandations sont effectuées, nous posons les notations suivantes :

– a est l'utilisateur actif

– $r_m \in R$ est une ressource où $R = \{r_1, r_2, \ldots, r_l\}$ est l'ensemble des ressources connues par le système

– $sa_t = (r_{a1}, r_{a2}, \ldots, r_{at})$ est la session active de a

– $S_t(r_m)$ est le score de la ressource r_m lorsque le système a connaissance des t dernières ressources consultées par l'utilisateur (ici sa_t).

– $RAS = \{ras_1, \cdots, ras_R\}$ est l'ensemble des règles d'association séquentielles du modèle

– Une règle ras_j est composée d'un antécédent $ant(ras_j)$ et d'une ressource conséquent $cons(ras_j)$

– La confiance d'une règle ras_j est notée $confiance(ras_j)$

– RAS_t est l'ensemble des règles correspondant à sa_t, la session composée des t dernières ressources consultées par l'utilisateur.

Pour évaluer le score de chaque ressource possible r_m, le système a deux possibilités, soit il parcourt la liste RAS et recherche celles dont l'antécédent se trouve dans sa_t, soit il extrait toutes les sous-séquences de sa_t et il recherche si elles sont présentes dans $ant(RAS)$. Au vu du nombre d'éléments dans RAS, il sera moins coûteux de choisir la deuxième solution.

Dans ce cas, le système doit dans un premier temps générer l'ensemble des sous-séquences de sa_t, puis trouver l'ensemble des règles RAS_t dont l'antécédent est égal à une sous-séquence de sa_t, et enfin extraire la confiance de ces règles puis en déduire le score de chaque ressource.

Trouver l'ensemble des sous-séquences de sa$_t$. Nous noterons $SubSeq$ la fonction qui, à une session sa_t lui associe la liste de ses sous-séquences. L'ensemble des sous-séquences résultantes sera noté $ss_{sa_t} = SubSeq(sa_t)$.

Trouver l'ensemble RAS$_t$. Pour chaque sous-séquence $ss_i \in ss_{sa_t}$, le système recherche si elle est égale à l'antécédent d'une RAS, c'est-à-dire une sous-séquence telle que $ss_i \in ant(RAS)$ où $ant(RAS)$ est, par abus de notation, l'ensemble des antécédents de RAS. Le résultat de cette étape est l'ensemble $RAS_t \in RAS$ telles que l'antécédent est égal à une sous-séquence $ss_i \in ss_{sa_t}$.

Evaluer le score de chaque ressource. Pour évaluer le score de chaque ressource $S_t(r_m)$, la confiance de chaque règle $ras_j \in RAS_t$ est utilisée. Etant donné que plusieurs règles de RAS_t peuvent avoir la même ressource conséquent, une ressource peut avoir plusieurs scores candidats. Plusieurs stratégies sont utilisées dans la littérature, choisissant soit le score maximal ou la somme de tous les scores (Brun *et al.,* 2009).

Lorsque la somme de tous les scores est utilisée, le score de r_m sera égal à la somme de toutes les confiances des règles de RAS_t dont le conséquent est r_m, il est calculé de la façon suivante :

$$S_t(r_m) = \sum_{j \in RAS_t \ \texttt{tq} \ cons(j)=r_m} confiance(j) \quad [1]$$

Lorsque le score maximal est utilisé, le score de r_m sera égal au maximum des confiances des règles dans RAS_t dont le conséquent est r_m, il sera calculé de la façon suivante :

$$S_t(r_m) = \max_{j \in RAS_t \ \texttt{tq} \ cons(j)=r_m} confiance(j) \quad [2]$$

Nous supposons que RAS est stocké sous forme d'arbre (cf. section 4.2). On supposera alors qu'atteindre un noeud fils dans un arbre, lorsque l'on est placé sur un

noeud a un coût de 1 (Nakagawa *et al.,* 2003). Dans ce cas, rechercher une sous-séquence ss_i dans un arbre se fait avec un coût de $O(|ss_i|)$ où $|ss_i|$ est la longueur de la sous-séquence ss_i.

Etant donné que dans une session active de taille t, il y a $2^t - 1$ sous-séquences possibles de taille entre 1 et t, faire correspondre la session active (et donc les sous-séquences) avec l'ensemble des RAS, se fait en $2^t - 1$ recherches. De plus, sachant que le coût d'une recherche est dépendant de sa taille, le coût total de la recherche sera égal à $t \cdot 2^{t-1}$. Cette valeur est d'ailleurs sous-estimée car la mise à jour du score de chaque ressource r_m est coûteux (Equations [1] et [2]).

Dans le cadre de la navigation Web, du e-commerce ou du m-commerce, la session active peut être très longue. Par exemple, dans un supermarché, il n'est pas rare qu'une session atteigne la taille de 30 produits, ce qui mène à plus de $1\ milliard$ de recherches dans l'arbre et donc plus de $16\ milliards$ d'opérations dans l'arbre. Le nombre d'opérations accroît exponentiellement avec la taille de la session.

Evidemment, lorsque l'on utilise une fenêtre glissante, le nombre de recherches est plus faible puisque la taille de la fenêtre est moindre. Cependant, nous avons montré (Bonnin *et al.,* 2009) que les recommandations les plus précises étaient obtenues avec une fenêtre glissante de 10 éléments (dans un cas de navigation sur un intranet), ce qui correspond à environ $5k$ opérations.

En conclusion, la génération de recommandations en exploitant des règles d'association séquentielles est une tâche complexe. Lorsque le système de recommandation est implanté sur un serveur, cela peut ne pas poser de problèmes de temps de calcul. A l'opposé, lorsque le système est implanté sur un client, un mobile par exemple dans le cadre du m-commerce, le temps de calcul requis peut en être grandement affecté, et les recommandations ne seront probablement pas présentées à l'utilisateur au moment où celui-ci prend en main un nouveau produit. La satisfaction de l'utilisateur en sera réduite.

4.4. *Des recommandations incrémentales*

Dans cette section, nous nous focalisons sur la réduction de la complexité du calcul des recommandations dans le but d'obtenir des recommandations en un temps minimum, et ainsi satisfaire le client.

Rappelons que le système de recommandation exploite la séquence des produits par lesquels le client a été intéressé. A un temps t, les recommandations proposées à l'utilisateur ne sont pas indépendantes des recommandations qui lui ont été faites au temps $t - 1$. En effet, la session exploitée par le système de recommandation au temps t est partiellement identique à celle exploitée au temps $t-1$, la seule différence se situant sur le dernier produit que l'utilisateur a pris en main (r_t). En partant de ce constat, nous proposons d'exploiter les calculs effectués au temps $t - 1$ pour le

calcul des nouvelles recommandations au temps t. Nous proposons donc un **système de recommandation incrémental**.

Nous introduisons maintenant deux nouvelles notations avant d'expliquer en détails les améliorations que nous proposons :

– $right(ras_j)$ représente le dernier élément de l'antécédent de la règle ras_j

– $left(ras_j)$ est la suite des éléments de l'antécédent de ras_j auquel on a supprimé le dernier élément ($right(ras_j)$).

La diminution de la complexité de calcul des recommandations au temps t que nous proposons va consister en trois améliorations :

– Exploiter le score de chaque ressource au temps $t-1$

– Exploiter la liste des règles RAS_{t-1}

– Exploiter le stockage de RAS sous forme d'arbre

Exploiter le score de chaque ressource au temps précédent. Pour cette étape nous proposons d'exploiter le fort recouvrement entre les deux sessions sa_{t-1} et sa_t. A un temps $t-1$, la session active est la suivante : $sa_{t-1} = (r_{a1}, r_{a2}, \ldots, r_{at-1})$. Le score $S_{t-1}(r_m)$ de chaque ressource r_m est connu. A ce même temps $t-1$, la liste RAS_{t-1} des règles correspondant à la session active a été construite en exploitant la suite de ressources de sa_{t-1}. Sachant que sa_{t-1} et sa_t sont en partie égales, nous pouvons en déduire qu'au temps t la liste RAS_t des règles correspondant à sa_t est composée de

– l'ensemble des règles de la liste RAS_{t-1}

– auquel sont ajoutées des nouvelles règles ras_j dont la première partie du corps est incluse dans la session sa_{t-1} ($left(ras_j) \in ss_{sa_{t-1}}$) et dont le dernier élément de l'antécédent est la nouvelle ressource consultée r_t ($right(ras_j) = r_t$)

Le système ayant déjà recherché l'ensemble de règles RAS_{t-1}, il ne lui reste plus qu'à rechercher l'ensemble de nouvelles règles.

Etant donné que l'exploitation, au temps $t-1$, de la liste RAS_{t-1} a mené au score $S_{t-1}(r_m)$, alors celui-ci sera mis à jour en exploitant l'ensemble de nouvelles règles. La complexité de la génération des recommandations en sera grandement réduite.

Rappelons ici qu'à chaque temps t, seules les ressources avec les plus hauts scores sont recommandées, mais que le score de chaque ressource est calculé.

Une première façon d'évaluer le score $S_t(r_m)$ de chaque ressource r_m au temps t peut donc se faire comme présenté ci-dessous, où l'équation [3] représente le cas où la somme des confiances est utilisée et l'équation [4] le cas où le maximum des confiances est utilisé.

$$S_t(r_m) = S_{t-1}(r_m) + \sum_{\substack{j \in RAS \\ \texttt{tq}\ left(j) \in ss_{sa_{t-1}} \\ \texttt{et}\ right(j) = r_t}} confiance(j) \qquad [3]$$

$$S_t(r_m) = \max(S_{t-1}(r_m), \max_{\substack{j \in RAS \\ \texttt{tq}\ left(j) \in ss_{sa_{t-1}} \\ \texttt{et}\ right(j) = r_t}} confiance(j)) \quad [4]$$

Le temps de calcul de $S_t(r_m)$ est donc diminué puisqu'une partie des recherches a été déjà effectuée au temps précédent.

Exploiter la liste des règles RAS_{t-1}. Sachant que les règles RAS ont été générées avec un algorithme incrémental, alors si une règle ras_n composée d'une suite de n ressources existe, alors il existe également dans RAS une règle ras_{n-1} composée d'une suite de $n-1$ ressources telle que $ant(ras_{n-1}) = left(ras_n)$. En partant de cette information, si à un temps t, une règle ras_t correspond à la session active sa_t, et telle que $right(ras_t) = r_t$ alors obligatoirement au temps $t-1$ une ras_{t-1} telle que $ant(ras_{t-1}) = left(ras_t)$ correspondait à sa_{t-1}.

Dans ce cas, la recherche du deuxième ensemble de règles pourra être optimisée puisque cette recherche va pouvoir exploiter les règles de RAS_{t-1} : pour chaque règle de RAS_{t-1}, on recherche si il existe une règle ras_j telle que $left(ras_j) \in ant(RAS_{t-1})$ et $right(ras_j) = r_t$). En effet, si une sous-séquence ss_i de sa_{t-1} ne correspondait à aucune règle, alors au temps t, il est inutile de rechercher une règle dont la partie gauche de l'antécédent correspond à une sous-séquence qui inclut ss_i.

L'évaluation du score de chaque ressource r_m se fera donc de la façon suivante. Lorsque la somme de tous les scores est utilisée, le score de r_m sera égal à

$$S_t(r_m) = S_{t-1}(r_m) + \sum_{\substack{j \in RAS \\ \texttt{tq}\ left(j) \in ant(RAS_{t-1}) \\ \texttt{et}\ right(j) = r_t}} confiance(j) \quad [5]$$

Lorsque le score maximal est utilisé , le score de r_m sera calculé de la façon suivante :

$$S_t(r_m) = \max(S_{t-1}(r_m), \sum_{\substack{ras_j \in RAS \\ \texttt{tq}\ left(ras_j) \in ant(RAS_{t-1}) \\ \texttt{et}\ right(ras_j) = r_t}} confiance(ras_j)) \quad [6]$$

Ici, le temps de calcul est encore diminué puisque les règles sont maintenant recherchées en s'aidant de l'ensemble restreint RAS_{t-1}.

Exploiter le stockage de RAS **sous forme d'arbre**

Soient deux règles ras_{t-1} et ras_t telles que l'antécédent de ras_{t-1} est inclut dans l'antécédent de ras_t et de $left(ras_t) = ant(ras_{t-1})$. La seule différence entre les deux antécédents de ces deux règles est donc le dernier élément de ras_t. Dans l'arbre où les RAS sont stockées, les deux règles ras_{t-1} et ras_t sont stockées dans la même

branche, ras_t étant même un noeud fils de ras_{t-1}. Partant de cette information, nous proposons de mémoriser à chaque temps $t-1$, l'ensemble des règles correspondant à la session active. Concrètement, un pointeur sur chaque règle (un noeud) de RAS_{t-1} est mémorisé. Au temps t, ces pointeurs, ainsi que l'information sur la dernière ressource consultée sont utilisés pour chercher si r_t est un noeud fils d'un des pointeurs (un noeud représentant une règle). Si c'est effectivement le cas, alors c'est que la règle correspondante existe. Les pointeurs correspondant aux règles RAS_t sont ensuite mis à jour.

En conclusion, nous pouvons dire que l'utilisation de la recommandation faite au temps précédent, de l'aspect incrémental des règles, alliés aux pointeurs, permet une large diminution du coût de calcul des scores des ressources au temps t par rapport à l'approche classique. Rappelons que dans l'approche classique, dans une session de taille t, $2^t - 1$ sous-séquences étaient possible et recherchées dans l'arbre. Dans notre proposition, le nombre de sous-séquences étudié est égal au nombre de sous-séquences étudiées au temps précédent $t - 1$. Pour une session de taille t, cette valeur est maintenant au plus égale à 2^{t-1}. Le nombre de recherches est donc divisé par au moins 2. En terme de coût total, le coût de recherche d'une sous-séquence était dépendant de sa taille. Dans notre proposition, son coût n'est plus que de 1 : elle se limite à la recherche d'un noeud fils. Le coût total dans une approche classique était de $t \cdot 2^{t-1}$, le coût total de notre approche n'est que de 2^{t-1}. Le coût total est donc divisé par t.

Dans le cas d'une session de taille 30, la génération des recommandations sera effectuée 30 fois plus vite. Les recommandation seront donc présentées à l'utilisateur plus tôt.

4.5. *Analyse*

L'algorithme que nous proposons ici a donc la caractéristique de réduire largement la complexité du calcul des recommandations. Contrairement à d'autres calculs de complexité, qui mesurent la complexité en moyenne ou au pire, les complexités que nous avons présentées sont des complexités exactes, c'est-à-dire que l'on connaît exactement la réduction de la complexité de notre approche. Il n'est donc pas utile d'effectuer de tests réels pour mesurer le gain effectif. De plus, la méthode incrémentale que nous avons proposée n'a aucune influence que les recommandations effectuées, leur qualité n'est en rien dégradée, elles sont identiques à celles effectuées avec l'approche classique. Pour cette raison nous n'avons pas jugé utile d'effecuer des expérimentations qui n'apporteraient aucune information concernant cette approche.

5. Conclusion

Les systèmes de recommandation ont rencontré un grand succès ces dernières années dans des applications web, en e-commerce et en m-commerce. Le m-commerce

est une application récente qui a émergé ces dernières années. Dans cet article, nous avons étudié l'exploitation des algorithmes classiquement utilisés dans les systèmes de recommandation, comme les règles d'association dans le cadre du m-commerce. Nous nous sommes focalisés sur les aspects respect de la vie privée, temps de calcul et qualité des recommandations. Dans le cadre du respect de la vie privée, nous choisissons d'implanter le système de recommandation sur le mobile, le souci majeur résultant de cette implantation est le temps de calcul des recommandations, dû à la capacité des mobiles.

Après avoir étudié et présenté la façon exacte dont les recommandations sont classiquement calculées dans le cadre de l'utilisation de règles d'association, nous concluons que cette étape est complexe. Nous avons ensuite proposé une nouvelle approche de la génération des recommandations qui permet d'obtenir des recommandations calculées incrémentalement : les recommandations à un temps donné sont fonction des recommandations faites au temps précédent. Cette nouvelle approche permet de réduire d'un facteur t, t étant la taille de la session courante, la complexité de la génération des recommandations. Cette réduction du temps de calcul a l'avantage de ne pas réduire la qualité des recommandations d'une approche classique.

Nous allons poursuivre ces travaux par l'étude d'autres algorithmes de data mining dans le but de les rendre eux aussi incrémentaux tout en conservant la qualité des recommandations.

6. Bibliographie

Adomavicius G., Tuzhilin A., « Toward the Next Generation of Recommender Systems : A Survey of the State-of-the-Art », *IEEE transactions on knowledge and data engineering*, vol. 17, n° 6, p. 734-749, 2005.

Agrawal R., Imielinski T., Swami A., « Mining Association Rules Between sets of Items in Large Databases », *Proceedings of the ACM SIGMOD Conference on Management od Data*, p. 207-216, 1993.

Anand S., Mulvenna M., Chevalier K., *Web Mining : From Web to Semantic Web*, Springer Berlin / Heidelberg, chapter On the Deployment of Web Usage Mining, p. 23-42, 2004.

Bonnin G., Brun A., Boyer A., « A Low-Order Markov Model integrating Long-Distance Histories for Collaborative Recommender Systems », *Proceedings of the ACM International Conference on Intelligent User Interfaces (IUI'09)*, Sanibel Islands, USA, p. 57-66, february, 2009.

Bozdogan H., *Statistical Data Mining and Knowledge Discovery*, Chapman & Hall/CRC, 2004.

Breese J., Heckerman D., Kadie C., « Empirical Analysis of Predictive Algorithms for Collaborative Filtering », *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, p. 43-52, 1998.

Brun A., Boyer A., « Towards Privacy Compliant and Anytime Recommender Systems », *In Proceedings of the E-Commerce and Web Technologies Conference (EC-Web09)*, p. 276-287, 2009.

Burke R., Hammond K., Cooper E., « Knowledge-based navigation of complex information spaces », *Proceedings of the 13th National Conference on Artificial Intelligence*, Menlo Park, Canada, p. 462-468, 1996.

Goldberg D., Nichols D., Oki B., Terry D., « Using collaborative filtering to weave an information tapestry », *Communications of the ACM*, vol. 35, n° 12, p. 61-70, 1992.

Han J., Kamber M., *Data Mining : Concepts and Techniques*, The Morgan Kaufmann Series in Data Management Systems, 2001.

Hosseini-Pozveh M., Nematbakhsh M., Movahhedinia N., « A multidimensional approach for context-aware recommendation in mobile commerce », *International Journal of Computer Science and Information Security*, 2009.

Hu W., Yeh J., Lee S., « Adaptive Web browsing using Web mining technologies for Internet-enabled mobile handheld devices », *Proceedings of the 16th Information Resources Management Association (IRMA 2006) International Conference*, 2006.

Huang Y., Kuo Y., Chen J., Jeng Y., « NP-miner : A real time recommendation algorithm by using web usage mining », *Knowledge-Based Systems*, vol. 19, p. 272-286, 2006.

Lee S., « A Mobile Application of Client Side Personalization Based on WIPI Platform », *Proceedings of the Computational and Information Science Conference (CIS04)*, p. 903-909, 2004.

Mobasher B., Dai H., Luo T., Nakagawa M., « Using Sequential and Non-Sequential Patterns for Predictive Web Usage Mining Tasks », *Proceedings of the IEEE International Conference on Data Mining (ICDM'2002)*, 2002.

Nakagawa M., Mobasher B., « Impact of Site Characteristics on Recommendation Models Based On Association Rules and Sequential Patterns », *Proceedings of the IJCAI'03 Workshop on Intelligent Techniques for Web Personalization*, 2003.

Pazzani M., Billsus D., *The Adaptive Web*, Springer Berlin / Heidelberg, chapter Content-Based Recommendation Systems, p. 325-341, 2007.

Sarwar B., Karypis G., Konstan J., Riedl J., « Analysis of Recommendation Algorithms for E-Commerce », *ACM Conference on Electronic Commerce*, 2000.

Srikant R., Agrawal R., « Mining Sequential Patterns : Generalizations and Performance Improvements », *Proceedings of the 5th International Conference on Extending Database Technology*, p. 3-17, 1996.

Tarasewich P., « Designing mobile commerce applications », *Communications of the ACM*, vol. 46, n° 12, p. 57-60, 2003.

Tewari G., Youll J., Maes P., « Personalized location-based brokering using an agent-based intermediary architecture », *Decision Support Systems*, vol. 34, n° 2, p. 127-137, 2003.

Tveit A., « Peer-to-peer based recommendations for mobile commerce », *International Workshop on Mobile Commerce, Proceedings of the 1st international workshop on Mobile commerce*, p. 26-29, 2001.

Wan Z., « Personalized Tourism Information System in Mobile Commerce », *IEEE International Conference on Management of e-Commerce and e-Government*, p. 387-391, 2009.

Xiao Y., Dunham M., « Efficient mining of traversal patterns », *Data and Knwoledge Engineering*, vol. 39, n° 2, p. 191-214, 2001.

Zenebe A., Ozok A., Norcio A., « Personalized Recommender Systems in e-Commerce and m-Commerce : A Comparative Study », *Proceedings of the 11th International Conference on Human-Computer Interaction*, 2005.