Vers un Filtrage Collaboratif Distribué : le modèle RSB

Sylvain Castagnos, Anne Boyer, François Charpillet

Vers un Filtrage Collaboratif distribué : le modèle RSB

S. Castagnos[†] A. Boyer[†] F. Charpillet[†]
castagno@loria.fr boyer@loria.fr charp@loria.fr

[†] Laboratoire Lorrain de Recherche en Informatique et Applications
Campus Scientifique, B.P. 239
54506 Vandœuvre-les-Nancy Cedex

Résumé :
Le terme de filtrage collaboratif [7] désigne les techniques utilisant les goûts connus d'un groupe d'utilisateurs pour prédire la préférence inconnue d'un nouvel individu. La particularité des procédés actuels de filtrage collaboratif est d'être centralisée. La problématique scientifique consistait donc à trouver un moyen de distribuer les calculs, afin d'assurer le passage à l'échelle pour des dizaines de milliers d'individus, ou encore préserver l'anonymat des utilisateurs (les données personnelles restent du côté client). L'impact d'un modèle combinant plusieurs méthodes existantes pour répartir les tâches entre le serveur et les postes clients est examiné dans cet article. Un partenariat avec la société ASTRA permet de recourir aux données de leur base et d'effectuer des tests grandeur nature pour vérifier l'efficacité des solutions proposées.

Mots-clés : Filtrage Collaboratif, Recommandations, Algorithme décentralisé, Réseaux Bayésiens, Coefficient de corrélation de Pearson.

Abstract:
The term of collaborative filtering [7] denotes techniques using the known tastes of a group of users to predict the unknown preference of a new user. The distinctive feature of current collaborative filtering processes is to be centralized. The scientific problems have consisted in finding a way to distribute calculus, in order to provide scale for several ten thousands of people, or then to preserve anonymity of users (personal data remain on client side). The impact of a model combining several existing methods to share out tasks between the server and users terminals is examined in this article. A partnership with the company ASTRA allows to turn to their database and to carry out life-sized tests in order to verify efficiency of proposed solutions.

Keywords: Collaborative filtering, Recommendations, Decentralized algorithm, Bayesian networks, Pearson correlation coefficient.

Remerciements

Nos recherches dans le domaine du filtrage collaboratif distribué s'inscrivent dans un contexte industriel. Aussi, nous tenons à remercier la société ASTRA pour avoir encourager ce travail.

Introduction

Avec le développement des nouvelles technologies de l'information et de la communication, ainsi que du *e-commerce*, la taille des bases de données s'est fortement accrue. Les utilisateurs ne peuvent pas consulter la quantité considérable de documents (également appelés « ressources ») pour repérer, en un temps raisonnable, les informations les préoccupant. Ils se contentent donc d'accéder aux quelques liens Internet qui leur sont familiers. Les systèmes de recommandations, de plus en plus répandus sur des sites tels que la Fnac[1] ou Amazon[2], fournissent aux clients des documents susceptibles de les intéresser mais qu'ils n'auraient pas forcément consultés spontanément. Ces procédés d'investigation requièrent des techniques de filtrage collaboratif. Concrètement, cela revient à identifier l'utilisateur courant à un groupe de personnes ayant les mêmes goûts et, ce, en fonction de ses préférences et de ses consultations passées. Ce système part du principe selon lequel les utilisateurs ayant apprécié les mêmes documents ont les mêmes centres d'intérêt. Il est ainsi possible de prédire les données susceptibles de répondre aux attentes des clients en profitant de l'expérience d'une population similaire.

La particularité des techniques existantes du filtrage collaboratif est d'être centralisée. Si la recherche des plus proches voisins parmi quelques milliers de candidats en temps réel n'est plus un problème, le passage à des dizaines de milliers d'utilisateurs reste à résoudre. Par ailleurs, la centralisation des données est en contradiction avec les consignes de la **C**ommission **N**ationale de l'**I**nformatique et des **L**ibertés[3] (CNIL). La confidentialité des informations relatives aux utilisateurs constitue une obligation légale. Nous avons donc choisi d'étudier les apports d'un système distribué dans le cadre de cette problématique.

L'objectif de cet article est double. D'une part, familiariser le lecteur aux problèmes liés au filtrage collaboratif ; d'autre part, proposer un

[1] http ://www.fnac.fr
[2] http ://www.amazon.com
[3] http ://www.cnil.fr

nouvel algorithme susceptible de pallier les difficultés énoncées préalablement. En conséquence, la première partie dresse un état de l'art critique des algorithmes de filtrage collaboratif existants. Elle aboutit à la conclusion selon laquelle une autre voie de recherche est à imaginer. La deuxième partie décrit notre modèle distribué, baptisé ***R**ecTree **S**EM **B**ayesian algorithm* (RSB). La solution adoptée consiste à combiner des techniques de filtrage existantes pour répartir les tâches entre le client et le serveur. Les avantages et inconvénients du modèle sont exposés dans une troisième partie. Nous évoquerons, en conclusion, les perspectives de recherche envisagées.

1 État de l'art

Le filtrage collaboratif utilise les comportements connus d'une population pour prévoir les futurs agissements d'un individu. Il s'agit, dans un premier temps, d'observer l'attitude du sujet[4] dans un contexte donné. Certains systèmes de recommandations mesurent l'intérêt suscité par une ressource sur la base de critères explicites tels que les votes[5]. D'autres disposent de données implicites comme, par exemple, le temps de consultation d'une page. Ces renseignements permettent ensuite de rechercher, par comparaison, les utilisateurs ayant des comportements similaires.

Les premières méthodes de filtrage collaboratif étaient purement statistiques. Une base de données contenaient les votes des utilisateurs afin d'identifier les documents les plus pertinents. Mais l'accroissement du nombre de ressources a rendu leur évaluation trop laborieuse pour les lecteurs. Le filtrage collaboratif doit aujourd'hui tenir compte de données incomplètes, voire manquantes, concernant les votes sur les items. C'est pourquoi David PENNOCK, Eric HORVITZ et C. Lee GILES jugent les méthodes purement statistiques inadaptées [12]. John BREESE, David HECKERMAN et Carl KADIE [2] ont identifié, parmi les techniques existantes, deux classes majeures d'algorithmes pour résoudre ce problème : les algorithmes basés sur la mémoire et ceux basés sur un modèle.

[4]Dans la suite de cet article, la personne qui se connecte sera qualifiée d'« utilisateur courant ».

[5]Les utilisateurs peuvent attribuer à chaque document lu une note allant, en général, de 1 à 7 ou de 1 à 10.

Les algorithmes basés sur la mémoire maintiennent une base de données des votes de tous les utilisateurs. Un score de similarité (RESNICK [13], MAES [14] ou BREESE [2]) est déterminé entre l'utilisateur courant et chacun des autres membres de la base. Chaque prédiction entraîne ensuite un calcul sur l'ensemble de cette source de données. L'influence d'un individu y est d'autant plus forte que son degré de similarité avec l'utilisateur courant est grand.

Ces techniques basées sur la mémoire présentent l'avantage d'être très réactives, en intégrant immédiatement au système les modifications des profils utilisateurs. BREESE *et alii* [2] s'accordent toutefois à trouver leur passage à l'échelle problématique : si ces méthodes fonctionnent bien sur des exemples de tailles réduites, il est difficile de passer à des situations caractérisées par un grand nombre de documents ou d'utilisateurs. En effet, la complexité des algorithmes en temps et en mémoire est beaucoup trop importante pour les grosses bases de données.

Les algorithmes basés sur un modèle constituent une alternative au problème de complexité combinatoire. Dans cette approche, le filtrage collaboratif peut être vu comme le calcul de la valeur attendue d'un vote, compte tenu des préférences de l'utilisateur courant. Ces algorithmes créent des modèles descriptifs corrélant les individus, les ressources et les votes associés via un processus d'apprentissage. Les prédictions sont ensuite inférées depuis ces modèles revêtant, le plus souvent, la forme de réseaux bayésiens. Ces derniers sont priviligiés pour leur capacité à mêler les probabilités issues d'un traitement statistique de retour d'expérience et les probabilités subjectives. UNGAR et FOSTER [15] ont également imaginé la possibilité de classifier les utilisateurs et les ressources en groupes. Pour chaque catégorie d'utilisateurs, il s'agit alors d'estimer la probabilité qu'une ressource soit choisie.

Selon PENNOCK *et alii* [11], les algorithmes basés sur un modèle minimisent le problème de la complexité algorithmique en mémoire. Par ailleurs, ils perçoivent dans ces modèles une valeur ajoutée au-delà de la seule fonction de prédiction : ils mettent en lumière certaines corrélations dans les données, proposant ainsi un raisonnement intuitif pour les recommandations ou rendant simplement les hypothèses plus ex-

plicites. Cependant, ces méthodes ne sont pas assez dynamiques et elles réagissent mal à l'insertion de nouveaux contenus dans la base de données. De plus, elle nécessite une phase d'apprentissage à la fois pénalisante pour l'utilisateur[6] et coûteuse en temps de calcul pour les grosses bases de données.

La difficulté principale du filtrage collaboratif reste donc le passage à l'échelle des systèmes. L'orientation actuelle consiste à scinder de manière explicite les calculs dits « *on-line* » de ceux dits « *off-line* ». Ceci aboutit au développement d'algorithmes hybrides (AGGARWAL [1], GOLDBERG [7], PENNOCK [11]). Ces solutions tirent partie des deux grandes familles évoquées plus haut pour réduire le nombre de calculs « *on-line* » et augmenter le nombre de calculs « *off-line* ». Cette séparation permet aux utilisateurs d'obtenir une réponse dans un délai raisonnable, puisqu'ils n'ont plus besoin d'attendre le traitement complet des données par le système. Toutefois, les calculs de prédiction effectués hors connexion sont très lourds et restent incompatibles avec un passage à grande échelle. De plus, les techniques hybrides existantes restent centralisés ce qui est contraire aux principes de protection de la vie privée imposés par la CNIL. L'algorithme proposé dans cet article est une nouvelle approche hybride, combinant les avantages des méthodes basées sur la mémoire et celles basées sur un modèle pour distribuer le travail entre le serveur et les postes clients. Soulignons, au passage, qu'un autre avantage d'une séparation des calculs de cette manière est de contourner le brevet américain[7] définissant le procédé de filtrage collaboratif centralisé dans le cadre d'une transmission satellitaire. Ceci est particulièrement intéressant dans le contexte de notre collaboration avec ASTRA.

2 L'algorithme RSB

L'architecture de l'algorithme RSB est représentée sur la figure 1. Le sigle RSB reflète donc l'association de l'algorithme de classification **R**ecTree (scindé en deux parties, respectivement du côté serveur et du côté client), du procédé d'apprentissage **S**EM et d'une méthode de filtrage collaboratif à base de réseaux **B**ayésiens.

[6] Le système de recommandations ne pourra pas fournir de documents pertinents dès les premières requêtes.

[7] *United States Patent number 5,790,935* daté du 4 Août 1998.

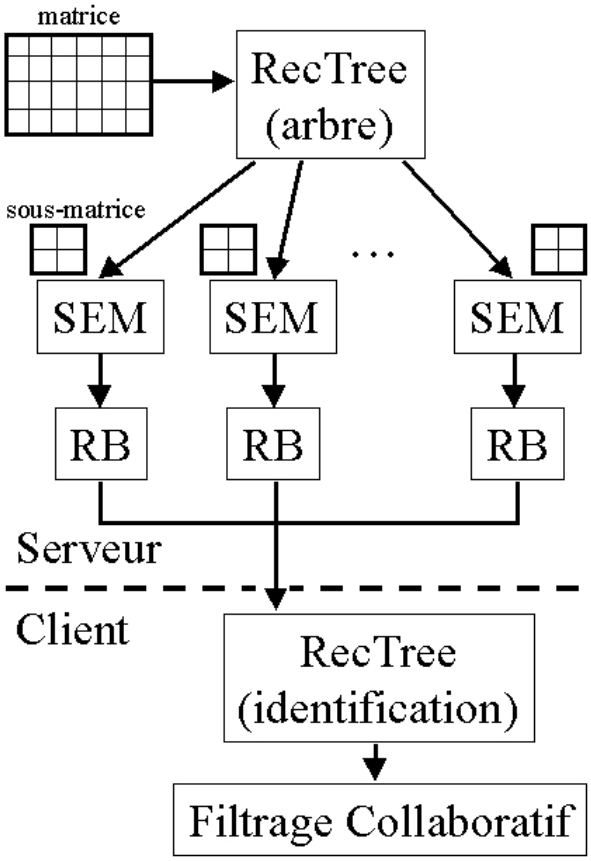

FIG. 1 – Architecture de l'algorithme RSB.

Afin de distribuer le système, la partie côté serveur a été dissociée de celle côté client. Le serveur dispose, en entrée, de la matrice des votes des utilisateurs et de la base de données contenant les sites et les descripteurs. De cette manière, le serveur ne dispose d'aucune information relative à la population, hormis les votes envoyés anonymement. Les préférences des utilisateurs sont stockées dans le profil sur les postes clients. Le critère de confidentialité est donc bien respecté.

De plus, il s'est agi de greffer, sur le groupe restreint obtenu, un deuxième système de filtrage collaboratif basé sur les réseaux bayésiens. La première passe avec RecTree n'avait donc pas pour objectif de faire du filtrage collaboratif, mais de réduire la quantité des données à traiter. Les calculs « *off-line* » de RecTree permettent la construction de profils d'utilisateurs type. De cette façon, il n'est plus nécessaire de considérer l'intégralité de la matrice des votes, mais uniquement les votes des individus appartenant au groupe de l'utilisateur courant. Cela réduit le nombre d'individus considérés, mais

également le nombre de ressources : il est inutile de conserver les documents qui n'ont été lus par aucun des membres du groupe.

2.1 RecTree

Le *clustering* hiérarchisé, également appelé `RecTree` [4], cherche à fractionner l'ensemble des utilisateurs en cliques.

Le tableau 1 propose un exemple de votes caractérisés par des entiers naturels allant de 1 à 10. Cette échelle de valeur est arbitraire. La précision de cette échelle doit être choisie par le concepteur du système, de telle sorte que les utilisateurs puissent faire la distinction entre les ressources qu'ils apprécient, qu'ils n'aiment pas ou qui les laissent indifférents.

TAB. 1 – Exemple de matrice des votes.

	R1	R2	R3	R4	R5
Utilisateur 1		6	7	6	
Utilisateur 2			4	7	7
Utilisateur 3		7	6	5	
Utilisateur 4	6	6			7

La figure 2 illustre les modalités d'organisation des groupes dans l'exemple ci-dessus. Les utilisateurs sont séparés en *clusters* selon la distance calculée entre eux.

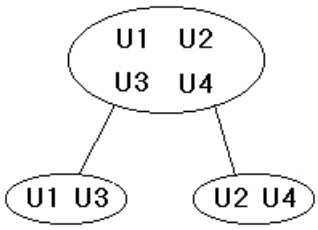

FIG. 2 – Hiérarchisation des utilisateurs.

L'algorithme `RecTree` est une méthode basée sur un modèle, dite de *clustering*. Elle se gère toutefois comme une approche basée sur la mémoire car toutes les informations sont nécessaires pour les calculs de similarité. Elle permet, dans le cadre de notre algorithme, de limiter le nombre d'individus à considérer dans le calcul de la prédiction. Ainsi, le temps de traitement par les réseaux bayésiens (cf. infra, Réseaux bayésiens, p. 6) sera plus court et les résultats seront potentiellement plus pertinents puisque les observations porteront sur un groupe plus proche de l'utilisateur courant [15]. Pour vulgariser, cela revient à demander leur avis[8] à un groupe de personnes ayant les mêmes goûts que l'utilisateur, plutôt que de consulter l'ensemble de la population. Chaque feuille de l'arbre `RecTree` correspond à un profil d'utilisateurs type.

La première étape consiste à associer à la racine de l'arbre la matrice globale des votes des utilisateurs par rapport aux ressources. Par la suite, l'ensemble des utilisateurs sont répartis en deux sous-groupes à l'aide de la méthode des plus proches voisins, également appelée `K-means` [8]. Cette dernière consiste, dans un premier temps, à choisir aléatoirement k centres dans l'espace de représentation utilisateurs/ressources. Dans le cas présent, le nombre k vaut 2, puisqu'il faut subdiviser la population en deux sous-ensembles. Chaque utilisateur est ensuite positionné dans le *cluster* de centre le plus proche (figure 3).

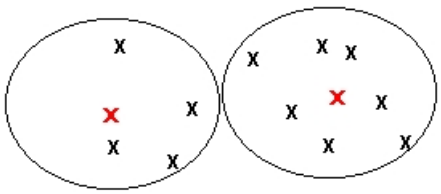

FIG. 3 – Espace de représentation utilisateurs/ressources.

La métrique utilisée pour déterminer la distance par rapport à ces centres est le coefficient de corrélation de Pearson [13] (cf. infra, formule 1, p. 5).

La littérature montre que le coefficient de corrélation de `Pearson` fonctionne bien [14], car il ne considère que les ressources communément évaluées par les utilisateurs comparés et ne tient pas compte des données manquantes.

[8] Bien sûr, le processus informatique est transparent pour les utilisateurs.

Formule du coefficient de corrélation de `Pearson` :

$$w(u_i, u_k) = \frac{\sum_{r \in R_i \cap R_k} (eval(u_i, r) - v)(eval(u_k, r) - v)}{\sqrt{\sum_{r \in R_i \cap R_k} (eval(u_i, r) - v)^2 \sum_{r \in R_i \cap R_k} (eval(u_k, r) - v)^2}} \quad (1)$$

Avec : $w(u_i, u_k)$ la distance entre u_i et u_k ;
$eval(u_i, r)$ l'évaluation de r par u_i ;
v la note moyenne de la ressource ;
R_i les ressources évaluées par u_i ;

Une fois les groupes de personnes ainsi formés, la position de l'isobarycentre est recalculée pour chaque *cluster* et l'opération est réitérée depuis le début jusqu'à obtenir un état stable (où les centres ne bougent plus après recalcul de leur position). L'algorithme des plus proches voisins est en $o(k^2n)$ pour k *clusters* et n individus. Une fois cette première subdivision effectuée, l'opération est renouvelée sur chacun des deux sous-groupes obtenus jusqu'à atteindre la profondeur de l'arbre souhaitée. Ainsi, plus on descend dans la structure et plus les *clusters* sont spécifiques à un certain groupe d'utilisateurs similaires. Par conséquent, plus on parcourt l'arbre en profondeur, plus les individus partagent le même avis concernant l'attribution d'une certaine note à un article donné. La construction de l'arbre s'opère en $o(n.log_2 n)$, où n est le nombre d'utilisateurs.

Par la suite, la phase d'identification de l'utilisateur courant à l'une des cliques s'opère du côté client en $o(2p)$, où p correspond à la profondeur de l'arbre. Ce dernier est construit de telle sorte que les cliques contiennent à peu près le même nombre d'individus pour une profondeur donnée.

2.2 SEM

On cherche ensuite à construire un réseau bayésien pour chacune des feuilles de l'arbre obtenue avec l'algorithme `RecTree`. Par définition, $B = (G, \theta)$ est un réseau bayésien si G = (X, E) est un graphe acyclique dirigé (**DAG**) dont les sommets représentent un ensemble de variables aléatoires X = $\{X_1, ..., X_n\}$, et si $\theta_i = [\mathbb{P}(X_i / X_{Pa(X_i)})]$ est la matrice des probabilités conditionnelles du noeud i connaissant l'état de ses parents $Pa(X_i)$ dans G [9]. Dans le cas étudié, chaque noeud correspond à une ressource et l'état du noeud à une valeur possible d'évaluation (cf. infra, figure 4). Comme toutes les méthodes par classifieur[9], l'utilisation des réseaux bayésiens nécessite d'introduire le concept de classe. Il faut en effet travailler sur une matrice de votes booléenne. Les classes « Aime » et « AimePas » ont été créées dans ce but. On considère par exemple qu'un utilisateur apprécie une ressource dès lors qu'il lui attribue une note supérieure ou égale à 6. Le tableau 2 transcrit la modification opérée dans le tableau 1.

TAB. 2 – Transformation en matrice booléenne.

	R1	R2	R3	R4	R5
U1 Aime	0	1	1	1	0
U1 AimePas	0	0	0	0	0
U2 Aime	0	0	0	1	1
U2 AimePas	0	0	1	0	0
U3 Aime	0	1	1	0	0
U3 AimePas	0	0	0	1	0
U4 Aime	1	1	0	0	1
U4 AimePas	0	0	0	0	0

La phase d'apprentissage consiste à rechercher les dépendances entre les ressources. Pour ce faire, nous avons recours à l'algorithme `Structural-EM`. Il présente l'avantage de gérer les données manquantes, ce qui est très intéressant dans ce contexte. Par ailleurs, ce procédé permet d'obtenir des résultats très proches des véritables structures[10] des réseaux bayésiens, quelle que soit la taille de la base d'ap-

[9] Il s'agit d'une subdivision des techniques de filtrage basées sur un modèle.

[10] Olivier FRANÇOIS et Philippe LERAY ont réalisé une étude comparative des algorithmes d'apprentissage dans le domaine des réseaux bayésiens [5]. Connaissant les structures des réseaux bayésiens, ils ont essayé de les réapprendre avec divers algorithmes.

prentissage [5]. Il s'agit d'une méthode itérative dont la convergence a été prouvée par Nir FRIEDMAN [6]. Il faut partir d'une structure initiale (pouvant être vide) pour estimer la distribution de probabilité des variables manquantes grâce à l'algorithme EM classique. Cela consiste à remplir les tables de probabilités conditionnelles à partir des paramètres. Puis, les données manquantes sont complétées en utilisant, par exemple, du bruit gaussien. Les probabilités sont enfin modifiées, afin de mieux correspondre à l'association des données observées et manquantes.

L'étape suivante consiste à calculer l'espérance d'un score (***B**ayesian **I**nformation **C**riterion*) par rapport à ces variables cachées pour tous les réseaux bayésiens du voisinage[11]. La structure ayant obtenu le meilleur score est ensuite choisie et l'opération est réitérée jusqu'à obtenir un maximum local.

2.3 Réseaux bayésiens

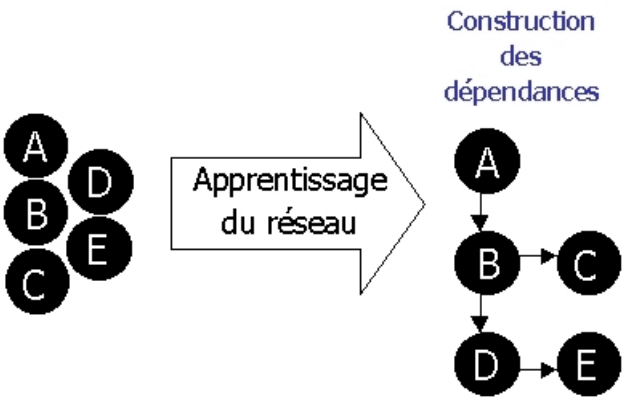

FIG. 4 – Construction des réseaux bayésiens.

Le module de filtrage collaboratif exploite les réseaux bayésiens obtenus du côté client. Une fois l'utilisateur identifié à un profil type à l'aide de RecTree du côté client, nous disposons du réseau bayésien spécifique à ce groupe de personnes. En chaque noeud, il est possible de construire un arbre de décision en fonction des prédécesseurs qui sont les plus à même de prédire l'état du noeud [2]. Ces arbres de décision sont donc une simple représentation des probabilités conditionnelles disponibles au niveau du noeud et établies lors de la phase d'apprentissage. Un parcours de l'arbre de décision en fonction de l'appréciation qui a été faite des prédécesseurs permet de prédire l'intérêt porté par l'utilisateur à la ressource du noeud courant. Dans l'exemple de la figure 4, si l'utilisateur a aimé les documents A et B, alors nous avons une certaine probabilité qu'il apprécie le document D.

Une approche plus classique consiste simplement à injecter les évidences dans les tables de probabilités conditionnelles, c'est-à-dire à utiliser les votes de l'utilisateur courant pour les personnaliser. Il ne reste plus qu'à consulter les probabilités que l'utilisateur courant apprécie une ressource non lue.

3 Discussion

L'algorithme RSB a été scindé en deux parties, respectivement côté client et côté serveur. De cette manière, la confidentialité des utilisateurs est respectée, dans la mesure où toutes les données personnelles sont stockées sur les postes clients, à l'exception des votes anonymes.

Par ailleurs, garantir un temps de réponse court du côté client est primordial. Mais il est difficile de se conformer à cet impératif si la base de données contient un grand nombre d'utilisateurs. En effet, afin d'assurer le caractère confidentiel des données relatives à l'utilisateur courant, la phase d'identification de ce dernier à un groupe doit se faire côté client. Cela signifie qu'il faut rapatrier les données relatives à la population sur le poste client. Pour éviter de surcharger le terminal utilisateur par envoi de la totalité de la matrice des votes, des profils d'utilisateurs type sont créés à l'aide de l'algorithme RecTree (cf. supra, 2.1 RecTree, p. 4). Le serveur se charge, dans un premier temps, de scinder l'ensemble des utilisateurs en cliques à partir de la matrice des votes (algorithme en $o(n.log_2 n)$). Puis la structure de l'arbre est envoyée au client où a lieu la phase d'identification de l'utilisateur courant à un groupe (algorithme en $o(2p)$). De cette manière, le nombre d'individus à considérer est restreint, puisque seuls ceux qui appartiennent à la même clique que l'utilisateur courant sont conservés[12]. Le choix de l'algorithme

[11] Les voisins d'un réseau bayésien sont toutes les structures différant du graphe d'origine par une opération d'ajout, suppression ou inversion d'arc.

[12] Rappelons qu'il est possible de choisir la profondeur de l'arbre : si

RecTree a été motivé par le fait que :
– cette méthode était facilement divisible en deux parties, exécutables respectivement côté client et côté serveur ;
– la partie du calcul « *on-line* », à savoir l'identification de l'utilisateur à un groupe, se fait en $o(2p)$. Ainsi, le temps de réponse côté client ne sera pas pénalisé par l'emploi de cet algorithme.

Il aurait été possible de se contenter de l'analyse de l'algorithme RecTree. Toutefois, nous avons choisi d'ajouter, sur le groupe restreint obtenu, un deuxième système de filtrage collaboratif basé sur les réseaux bayésiens. En effet, la seule utilisation de RecTree ne suffit pas à pallier le problème de la taille de la base de données. Cette utilisation de l'algorithme RecTree permet de réduire la quantité de données qui doit être traîtée lors de la construction des profils type. Ainsi, il suffit de considérer uniquement les votes et ressources lues par les individus appartenant au groupe de l'utilisateur courant.

Le choix de la méthode de filtrage effective s'est porté sur les réseaux bayésiens. Ces derniers permettent notamment la modélisation des relations non-déterministes existantes entre les ressources. De plus, la grande majorité des calculs se fait de manière « *off-line* » pour structurer le modèle de documents et déterminer les probabilités conditionnelles. Seul le parcours de l'arbre de décision, au niveau du noeud à évaluer, nécessite un calcul « *on-line* ». La phase d'apprentissage du réseau bayésien (recherche des dépendances entre les ressources) reste un facteur limitant en terme de temps de traitement. Toutefois, les résultats obtenus restent bien supérieurs à un algorithme classique basé sur la mémoire par exemple. Par ailleurs, la méthode SEM choisie permet de gérer les données manquantes. Cela est intéressant dans la mesure où les utilisateurs ne votent en général que pour une infime partie des ressources mises à leur disposition.

L'algorithme RSB a une complexité, en terme de temps de calcul, bien moindre que la méthode de filtrage collaboratif basée sur la mémoire [14] imaginée par Pattie MAES et Upendra SHARDANAND. Cette technique avait été implantée au préalable au sein de notre plateforme de travail [10]. La base de données utilisée est celle de la société de diffusion de sites Web par satellites ASTRA[13], contenant les votes de plus de 80.000 individus pour 100 ressources.

Les calculs du côté serveur (RecTree et SEM) sont trop encore lourds pour être effectués en temps réel et doivent être renouvelés périodiquement. Cette façon de procéder peut certes augurer de légers écarts entre les derniers votes et les préférences prises en compte dans les calculs de prédiction. Toutefois, ces écarts restent minimes en raison du grand nombre d'utilisateurs. De plus, cette pratique assure la stabilité du système en cas d'ajout de documents, puisque ces derniers ne seront considérés qu'après réitération des calculs du côté serveur.

Perspectives

L'algorithme RSB effectue du filtrage collaboratif de façon distribué sur de grosses bases de données et dans un délai relativement court. Toutefois, il importe de mesurer la satisfaction des utilisateurs par rapport aux ressources que le système leur propose. Un retour sur la pertinence des sites suggérés permettrait de valider le modèle. A cette fin, nous sommes en train de finaliser des méthodes d'évaluation permettant de vérifier la pertinence des propositions faites par le système.

Afin d'obtenir un plus grand nombre de votes, une fonction d'aide basée sur la formule de Philip CHAN [3] (cf. infra, formule 2, p. 8) peut aussi être envisagée : elle se charge d'estimer les notes que l'utilisateur est susceptible d'attribuer à différents sites à partir de critères implicites (tels que le temps ou la fréquence de consultation d'une page[14]). Le formulaire de votes serait ainsi pré-rempli et soumis à approbation pour faire gagner du temps à l'utilisateur.

L'étude de l'attrait des architectures *Peer-to-Peer* constitue une perspective de recherche possible : le système pourrait se charger de faire communiquer plusieurs postes utilisateurs entre

cette dernière n'est pas excessive, les groupes étudiés sont plus petits que la population totale mais encore disparates et donc capables de suggérer des nouveautés à l'utilisateur.

[13] http ://www.ses-astra.com. Ils ont développé un logiciel du nom de Casablanca, utilisant la technologie Sat@once.

[14] Ce sont des informations facilement récupérables légalement et localement dans le navigateur Web du client.

Formule de Chan :

$$Interet(page) = Frequence(page) \,.\, (1 + EstFavori(page) + Duree(page) + Recent(page) + PourcentLiensVisites(page))$$

$$\text{Avec : } Duree(page) = max_{pages\ visitees}\left(\frac{duree\ passee\ sur\ page}{taille\ de\ la\ page}\right)$$

$$\text{Et : } Recent(page) = \frac{temps(derniere\ visite) - temps(debut\ log)}{temps(actuel) - temps(debut\ log)} \quad (2)$$

$Interet(page)$ doit être normalisé pour correspondre à l'échelle des votes.
$EstFavori(page)$ vaut 1 si la page fait partie des favoris de l'utilisateur et 0 sinon.
Enfin, $PourcentLiensVisites(page)$ correspond au nombre de liens visités divisé par le nombre de liens dans la page.

eux par ce biais pour former des groupes de personnes similaires. Enfin, un moyen d'améliorer le système serait de considérer le cache comme une boîte noire contenant une multitude de techniques de filtrage collaboratif. Le système se chargerait alors d'identifier tout seul la combinaison de méthodes la plus pertinente en fonction des données dont il dispose.

Références

[1] Charu C. Aggarwal, Joel L. Wolf, Kun-Lung Wu and Philip S. Yu. *Horting hatches an egg : a new graph-theoretic approach to Collaborative Filtering*. In Knowledge Discovery and Data Mining. pages 201-212, 1999.

[2] J. Breese, D. Heckerman and C. Kadie. *Empirical Analysis of Predictive Algorithms for Collaborative Filtering*. In *Proceedings of the fourteenth Annual Conference on Uncertainty in Artificial Intelligence* (UAI-98). San Francisco, CA, p. 43-52, July 1998.

[3] Philip Chan. *A non-invasive learning approach to building user profiles*. Web Usage Analysis and User Profiling. 1999.

[4] S. H. S. Chee, J. Han and K. Wang. *RecTree : An Efficient Collaborative Filtering Method'*. In *Proceedings 2001 Int. Conf. on Data Warehouse and Knowledge Discovery* (DaWaK'01). Munich, Germany, September 2001.

[5] O. François et P. Leray. *Etude comparative d'algorithmes d'apprentissage de structure dans les réseaux bayésiens*. In *Proceedings of RJCIA03*, plate-forme AFIA03. pages 167-180, 2003.

[6] Nir Friedman. *The Bayesian Structural EM Algorithm*. In Gregory F. Cooper and Serafín Moral editors, *Proceedings of the fourteenth Conference on Uncertainty in Artificial Intelligence* (UAI-98), Morgan Kaufmann Publishers. San Francisco, pages 129-138, 24-26 July 1998.

[7] Ken Goldberg, Theresa Roeder, Dhruv Huptan and Chris Perkins. *Eigentaste : a constant time Collaborative Filtering algorithm*. Technical Report M00/41. IEOR and EECS Departments, UC Berkeley, August 2000.

[8] J. L. Herlocker, J. A. Konstant, A. Borchers and J. Riedl. *An algorithmic framework for performing Collaborative Filtering*. In *Proceedings 1999 Conference of Research and Development in Information Retrieval*. Berkeley, CA, pages 230-237, August 1999.

[9] Cecil Huang and Adnan Darwiche. *Inference in Belief Networks : a Procedural Guide*. International Journal of Approximate Reasoning, Elsevier Science Inc. 1994.

[10] Régis Lhoste. *ArchimaJ, une plate-forme multi-agents pour la recherche d'informations*. Mémoire de DRT. Université Nancy 2, 25 Novembre 2003.

[11] David M. Pennock, Eric Horvitz, Steve Lawrence and C. Lee Giles. *Collaborative filtering by personality diagnosis : a hybrid memory- and model-based approach*. In *Proceedings of the sixteenth Conference on Uncertainty in Artificial Intelligence* (UAI-2000). Morgan Kaufmann Publishers. San Francisco, 2000.

[12] David M. Pennock, Eric Horvitz and C. Lee Giles. *Social choice theory and recommander systems : analysis of the axiomatic foundations of collaborative filtering*. In *Proceedings of the seventeenth National Conference on Artificial Intelligence*. July 2000.

[13] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, John Riedl. *GroupLens : an Open Architecture for Collaborative Filtering of Netnews*. In *Proceedings of ACM 1994 Conference on Computer Supported Cooperative Work*. Chapel Hill, North Carolina, 1994.

[14] Upendra Shardanand and Pattie Maes. *Social Information Filtering : algorithms for automating "word of mouth"*. In *Proceedings of CHI'95, Human factors in computing systems*. pages 210-217, 1995.

[15] L. Ungar and D. Foster. *Clustering Methods for Collaborative Filtering*. In *Fifteenth Workshop on Recommendation Systems*. July 1998.