RecoOLAP : Un système de recommandation de requêtes OLAP

Elsa Negre

Université Paris-Dauphine, LAMSADE
elsa.negre@dauphine.fr

Résumé. Une session d'analyse OLAP peut être définie comme une session interactive durant laquelle un utilisateur lance des requêtes pour naviguer dans un cube. Très souvent, choisir quelle partie du cube va être naviguée par la suite, et, de ce fait, concevoir la prochaine requête, est une tâche difficile. Dans cette démonstration, nous proposons un système de recommandation qui utilise ce que tous les utilisateurs du système OLAP ont fait pendant leurs précédentes explorations du cube afin de recommander des requêtes MDX à l'utilisateur.

1 Introduction

La recommandation de requêtes dans les bases de données a fait l'objet de nombreuses recherches, (Khoussainova et al. (2009), Chatzopoulou et al. (2009)), en particulier dans les systèmes OLAP (OnLine Analytical Processing) où l'utilisateur navigue interactivement dans un cube en lançant une séquence de requêtes (et plus particulièrement des requêtes MDX) sur un entrepôt de données, ce que nous appelons une session d'analyse (ou session) dans la suite de l'article. Ce processus est souvent pénible puisque l'utilisateur peut ne pas avoir d'idée sur ce que pourrait être la prochaine requête (Sarawagi (2000)).

Les systèmes de recommandation existants sont généralement classés selon deux catégories : les méthodes basées sur le contenu et celles basées sur le filtrage collaboratif (Adomavicius et Tuzhilin (2005)). Les méthodes basées sur le contenu recommandent à l'utilisateur des objets similaires à ceux qui l'ont intéressé dans le passé, tandis que les méthodes basées sur le filtrage collaboratif recommandent des objets qui ont intéressé des utilisateurs similaires. Au vu de la nature multi-utilisateur des entrepôts de données, dans nos travaux précédents (cf. Negre (2009) pour une vue d'ensemble), nous avons proposé des techniques issues du filtrage collaboratif pour recommander des requêtes OLAP à l'utilisateur. L'idée principale est de calculer une similarité entre la séquence de requêtes de l'utilisateur courant et les séquences de requêtes précédentes qui ont été enregistrées par le serveur.

Nous présentons, dans cette démonstration, notre cadre générique de génération de recommandations élaboré lors de nos travaux, appliqué sur des données synthétiques, qui permet de recommander des requêtes MDX, en utilisant le log du serveur, c'est-à-dire, l'ensemble des sessions précédentes sur le cube, et la séquence de requêtes de la session courante.

Cette démonstration est organisée comme suit : la section 2 présente notre cadre générique de recommandations de requêtes et la section 3 détaille l'architecture de notre système. Enfin, nous concluons section 4.

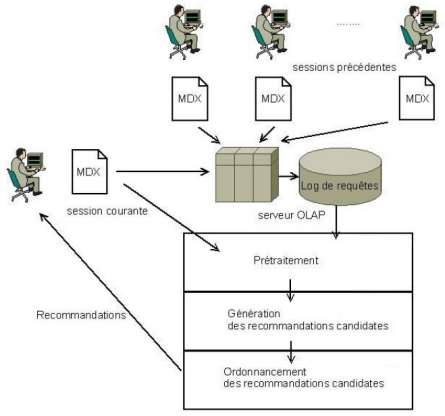

FIG. 1 – *Notre système : RecoOLAP*

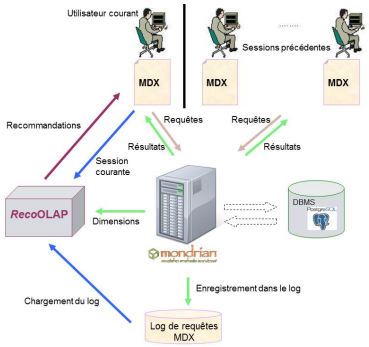

FIG. 2 – *Architecture du système RecoOLAP*

2 Présentation

Le but de notre système de recommandation est d'aider l'utilisateur à naviguer dans un cube de données en exploitant ce que les autres utilisateurs ont fait pendant leurs navigations précédentes. Notre cadre générique est fondé sur le processus suivant : (i) prétraitement du log, (ii) génération des recommandations candidates en commençant par trouver quelles sessions du log coïncident avec la session courante et ensuite, prédire ce que peut être la prochaine requête, (iii) ordonnancement des requêtes candidates en présentant à l'utilisateur la requête la plus pertinente en premier. Ce cadre est générique dans le sens où il n'impose pas une méthode particulière de prétraitement du log, de génération des requêtes candidates ou d'ordonnancement de celles-ci. Au lieu de cela, ces actions sont laissées comme paramètres du cadre qui peut être instancié de diverses manières afin de changer la méthode de calcul des recommandations. La figure 1 illustre le fonctionnement de notre système.

Notre système propose actuellement différents paramètres : (i) Deux fonctions de prétraitement : l'une utilisant l'algorithme des k-médoïdes (Kaufman et Rousseeuw (1987)), l'autre étant l'identité, (ii) Deux fonctions de matching : l'une permettant d'obtenir les sessions qui coïncident avec la session courante et à quelle position de la session du log a lieu cette coïncidence et l'autre renvoyant la (ou les) session(s) la (les) plus proche(s) de la session courante au sens d'une distance entre sessions et, cinq représentants de session : successeur, dernier, union, intersection ou médoïde, (iii) Deux méthodes d'ordonnancement des requêtes : l'une basée sur la proximité entre la requête à recommander et la requête qui représente la session courante (au sens d'une distance entre requêtes), l'autre basée sur le profil de l'utilisateur.
Quatre combinaisons particulières ont été proposées afin d'illustrer l'applicabilité de nos algorithmes génériques (cf. Negre (2009)) : ClusterH, ClusterSP, EdH et EdSP.

ClusterH : Le prétraitement est réalisé en utilisant l'algorithme des K-médoïdes (nous obtenons des classes de requêtes). La génération des recommandations est réalisée en cherchant des sous-séquences de la session courante prétraitée qui coïncident avec les sessions prétraitées du log puis en retournant le médoïde des classes qui succèdent à chaque sous-séquence similaire. L'ensemble des requêtes candidates ainsi obtenu est ensuite ordonné en fonction de

la proximité des requêtes avec la dernière requête de la session courante. Les distances utilisées pour les calculs de similarité sont la distance de Hausdorff et la distance de Hamming.

ClusterSP : Les distances utilisées sont la distance de Hausdorff et la distance basée sur le plus court chemin dans un graphe. Les autres paramètres sont similaires à ceux de ClusterH.

EdH : Le log n'est pas prétraité. La génération des recommandations est réalisée en calculant la distance de Levenshtein entre les sessions de requêtes du log et la session courante (nous gardons celles qui coïncident) puis en retournant la dernière requête de chaque session candidate. L'ensemble des requêtes candidates ainsi obtenu est ensuite ordonné en fonction de la proximité des requêtes avec la dernière requête de la session courante. Les distances utilisées pour les calculs de similarité sont la distance de Hausdorff et la distance de Hamming.

EdSP : Les distances utilisées sont la distance de Hausdorff et la distance basée sur le plus court chemin dans un graphe. Les autres paramètres sont similaires à ceux de EdH.

3 Implémentation

Dans cette section, nous présentons l'environnement $RecoOLAP$ développé afin de valider et de rendre opérationnel notre cadre générique de recommandation de requêtes. L'objectif est d'offrir un environnement recommandant à un utilisateur donné, des requêtes MDX en fonction d'une session d'analyse courante et d'un log de sessions de requêtes MDX.

Notre système de recommandation fonctionne de la manière suivante : à partir d'un log de sessions de requêtes MDX et d'une session courante de requêtes MDX, en appliquant notre algorithme de recommandation selon certains paramètres définis, le système propose un ensemble ordonné de requêtes MDX pouvant être recommandées à la suite de la session courante.

La Figure 2 illustre l'architecture de notre système. Premièrement, chaque requête lancée par un utilisateur sur le cube de données, stocké dans le SGBD PostgreSQL (PostgreSQL (2013)), via le serveur OLAP Mondrian (Pentaho Corporation (2009)) obtient une réponse. Deuxièmement, les sessions de requêtes des utilisateurs précédents ont été enregistrées dans un log de sessions de requêtes. La session de requêtes de l'utilisateur courant : la session courante, ainsi que les sessions de requêtes du log sont chargées dans l'application de génération de recommandations : RecoOLAP. Pendant le processus de recommandation, RecoOLAP accède aux informations contenues dans le serveur OLAP Mondrian. Enfin, le système retourne à l'utilisateur l'ensemble ordonné des recommandations.

Notons que l'application pour la recommandation de requêtes MDX est développée en Java sous JRE 1.6.0-27 avec Postgres 9.1.10 et Mondrian 3.3.0.14703. Tous les tests ont été réalisés avec un Core i5-2520M (2.5 Ghz $\times$ 4) à 8GB de RAM utilisant Linux Ubuntu 12.04.

4 Conclusion

RecoOLAP est un système de recommandation (collaboratif) de requêtes OLAP (Negre (2009)) qui aide l'utilisateur à avancer dans son analyse d'un cube de données. En effet, à partir d'un log de requêtes précédemment lancées par différents utilisateurs sur un cube de données et d'une session de requêtes courante lancée sur un même cube par un autre utilisateur, le système retourne un ensemble de requêtes recommandées pouvant faire suite à la session de requêtes courante.

Nous avons expérimenté notre cadre générique de génération de recommandations sur des données synthétiques (base de données *foodmart* fournie avec le moteur OLAP Mondrian) et les tests ont révélé que notre système donne des résultats plutôt satisfaisants : Les temps de génération d'une recommandation dépassent rarement la seconde (bien que le log puisse contenir env. 1000 requêtes et la session courante env. 10 requêtes), les requêtes recommandées générées sont de très bonne qualité puisque la F-mesure atteint 0.8 pour plus de 45% des sessions (Negre (2009)). Il est à noter que des tests ont été réalisés sur des données réelles (non présentées lors de la démonstration pour des raisons de confidentialité) et que les résultats de pertinence et de temps de génération sont similaires. Finalement, nous travaillons actuellement sur l'évaluation de la qualité de tels systèmes dans l'optique de comparer notre système de recommandation de requêtes OLAP à d'autres systèmes similaires.

Références

Adomavicius, G. et A. Tuzhilin (2005). Toward the next generation of recommender systems : A survey of the state-of-the-art and possible extensions. *IEEE TKDE 17*(6), 734–749.

Chatzopoulou, G., M. Eirinaki, et N. Polyzotis (2009). Query recommendations for interactive database exploration. In *SSDBM*, pp. 3–18.

Hamming, R. W. (1950). Error detecting and error correcting codes. *Bell System Technical Journal 29*(2), 147–160.

Hausdorff, F. (1914). *Grundzuge der Mengenlehre*. von Veit.

Kaufman, L. et P. Rousseeuw (1987). *Clustering by Means of Medoids*. Reports of the Faculty of Mathematics and Informatics. Faculty of Mathematics and Informatics.

Khoussainova, N., M. Balazinska, W. Gatterbauer, Y. Kwon, et D. Suciu (2009). A case for a collaborative query management system. In *CIDR*. www.crdrdb.org.

Levenshtein, V. (1966). Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Soviet Physics Doklady 10*, 707.

Negre, E. (2009). *Exploration collaborative de cubes de données*. Ph. D. thesis, Université François Rabelais, Tours, France.

Pentaho Corporation (2009). Mondrian open source OLAP engine. Available at http ://mondrian.pentaho.org/.

PostgreSQL (2013). Postgresql open source object-relational database system. Available at http ://www.postgresql.org/.

Sarawagi, S. (2000). User-adaptive exploration of multidimensional data. In *VLDB*.

Summary

An OLAP analysis session can be defined as an interactive session during which a user launches queries to navigate within a cube. Very often choosing which part of the cube to navigate further, and thus designing the forthcoming query, is a difficult task. In this demonstration, we propose a recommender system that uses what the OLAP users did during their former exploration of the cube as a basis for recommending MDX queries to the user.