Recommandation de chemins de navigation dans un cube OLAP

Rym Khemiri, Fadila Bentayeb

Laboratoire ERIC, Université de Lyon - Lyon 2
5 Avenue Pierre Mendès-France, 69676 Bron Cedex, France
(rym.khemiri, fadila.bentayeb)@univ-lyon2.fr
https://eric.ish-lyon.cnrs.fr/

Résumé. L'analyse OLAP (On-Line Analytical Processing), malgré son aspect navigationnel et exploratoire, ne permet pas de guider l'utilisateur vers les faits les plus pertinents lors de sa navigation au sein d'un cube de données. Ceci peut s'expliquer par le fait que l'OLAP ne tient pas compte des usages des utilisateurs, comme l'historique de ses requêtes de navigation. Pour pallier ce problème, nous proposons dans cet article NAPARE (NAvigation PAth REcommendation), un système de recommandation collaborative de chemins de navigation qui s'appuie sur l'historique des sessions d'analyse des utilisateurs et les chaînes de Markov.

1 Introduction

Dans un cube de données, extrait à partir d'un entrepôt de données, les données sont organisées de manière à permettre aux décideurs de les exploiter en utilisant les opérateurs OLAP. La navigation OLAP consiste alors en une interrogation interactive des données, en réalisant des analyses à travers de multiples passes (passant par exemple des données résumées à des données détaillées ou changeant même d'axe d'observation), successivement dans des niveaux de détail inférieurs permettant ainsi de produire les premiers résultats d'analyse pertinents pour l'aide à la décision. Ainsi, une session d'analyse typique sur un cube de données est une séquence de requêtes de navigation. Chaque requête d'une séquence est formulée sur la base des résultats de la requête précédente.

Cependant, lors de la navigation dans un cube de données, l'utilisateur n'a pas une connaissance a priori des parties du cube susceptibles d'être intéressantes pour lui. De ce fait, choisir les prochaines navigations devient une tâche difficile pour lui puisque plusieurs chemins de navigation se présentent à lui. Cela pourrait provoquer des temps de latence de l'analyse voire de conduire l'utilisateur dans une zone non pertinente, et de réduire ainsi les avantages de l'utilisation du système OLAP.

Le défi auquel nous nous intéressons dans cet article est de pouvoir guider l'utilisateur vers les faits les plus pertinents pour lui en utilisant un système de recommandation de chemins de navigation. Généralement, la plupart des systèmes de navigation existants souffrent d'un certain nombre de limitations Bentayeb et al. (2009). Le graphe de navigation nécessite généralement du hasard ou de l'expertise des utilisateurs, ou les deux à la fois pour trouver les bons

chemins menant vers les éléments recherchés. Les liens de navigation peuvent aussi conduire à des résultats vides.

Toutefois, dans un cube OLAP, les données sont modélisées suivant de multiples dimensions où les sous-cubes sont généralement représentés comme des éléments de treillis de cuboïdes Harinarayan et al. (1996). C'est pourquoi, nous utilisons la structure de treillis de cuboïdes pour décrire les chemins de navigation d'un utilisateur. Le graphe de navigation est donc le treillis de tous les cuboïdes. En effet, la structure de treillis de cuboïdes permet de visualiser tous les chemins de navigation d'un utilisateur et permet surtout de garder le séquencement logique de navigation. De ce fait, nous assimilons la notion de session d'analyse à un chemin de navigation dans le treillis de cuboïdes correspondant au cube OLAP. Dans une session d'analyse, l'utilisateur passe d'une requête d'analyse à une autre qui lui est directement liée formant ainsi un enchaînement de requêtes appelé chemin de navigation. La requête d'analyse suivante ne dépend que de la requête d'analyse en cours. De ce fait, afin de guider l'utilisateur vers un chemin de navigation pertinent pour lui, notre idée est d'utiliser les chaînes de Markov qui ont la propriété suivante : "Le futur ne dépend que de l'état présent". Autrement dit, il est possible de prédire l'état suivant avec la seule connaissance de l'état présent. Ainsi, à partir d'un point de navigation (nœud courant dans le treillis de cuboïdes), et en utilisant la propriété des chaînes de Markov, il devient possible de prédire le point de navigation suivant à l'utilisateur.

2 Concepts généraux

Dans cette section, nous donnons les notions fondamentales utilisées dans ces travaux à savoir la navigation OLAP, le treillis de cuboïdes et le modèle de Markov.

2.1 Navigation dans les cubes OLAP

L'analyse OLAP donne aux utilisateurs la possibilité d'analyser et d'explorer les données de manière interactive sur la base du modèle multidimensionnel. Bien que les utilisateurs d'outils de reporting jouent essentiellement un rôle passif, les utilisateurs OLAP sont en mesure de démarrer une session d'analyse complexe où chaque étape est le résultat de l'issue de l'étape précédente. La navigation peut alors être expliquée par la transition entre les différents états Dittrich et al. (2005).

La navigation est un terme utilisé pour décrire le processus employé par les utilisateurs pour explorer un cube de données de façon interactive, habituellement en utilisant un client OLAP graphique connecté à un serveur OLAP. Généralement, un utilisateur commence à analyser des données en sélectionnant une requête initiale. Cette requête est constituée d'un ensemble de dimensions ainsi que d'un ensemble de conditions de filtrage Khemiri et Bentayeb (2013). Ensuite, l'utilisateur modifie la requête de manière interactive en ajoutant ou en supprimant des colonnes (drill-down et roll-up), en ajoutant ou en supprimant les conditions de filtrage (slicing), en déplaçant des colonnes (dicing) et ainsi de suite.

2.2 Treillis de cube de données

Le concept du cube de données a été proposé la première fois par Gray et al. (1996) comme une généralisation de l'opérateur *Group By* de SQL pour répondre à l'enquête en ligne des

données de différents points de vue des utilisateurs. Cette analyse interactive et multidimensionnelle est habituellement accomplie par agrégations pré-calculées sur les données Sarawagi (2000). En effet, Harinarayan et al. (1996) proposent la modélisation des données dans de multiples dimensions où les vues OLAP sont généralement représentées comme des éléments de treillis. Le treillis de cube de données est un DAG (Directed Graph acyclique) dont les nœuds représentent des requêtes (ou cuboïdes) qui sont caractérisées par les attributs de la clause Group By. Les arêtes dénotent la relation de dérivabilité entre les cuboïdes. Par exemple, un cube de données avec les dimensions A_1, A_2, A_3 est présenté dans la Figure 1 (a) comme une structure en treillis. Généralement, dans le contexte OLAP, les dimensions sont organisées en hiérarchies de dimensions, qui peuvent être également représentées par un treillis. Par exemple, la Figure 1 (b) montre le treillis des attributs de trois dimensions A_1, A_2, A_3 où a_{ij} est le $j^{\text{ième}}$ niveau dans la hiérarchie de la dimension A_i. L'élément supérieur de chaque treillis est "all", ce qui signifie qu'il n'y a pas de regroupement pour cette dimension.

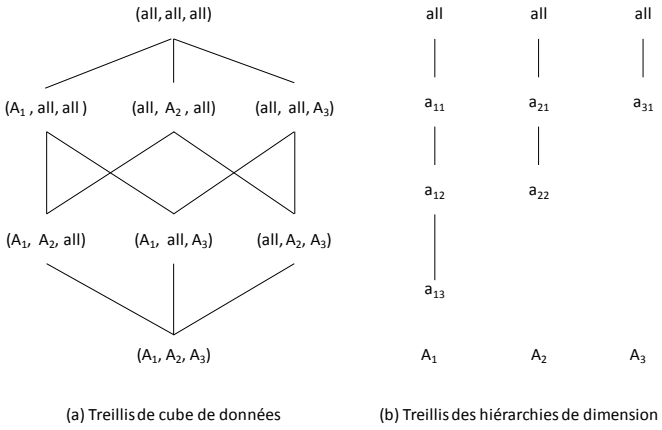

FIG. 1 – *Exemples de treillis de cubes OLAP*

Nous pouvons construire le treillis impliquant les hiérarchies qui représente l'ensemble des points de vue qui peuvent être obtenus en regroupant sur chaque combinaison d'éléments l'ensemble des hiérarchies de dimension. La Figure 2 montre le treillis qui combine le treillis de cube de données de la Figure 1 (a) avec les treillis de la hiérarchie des dimensions de la Figure 1 (b).

Par conséquent, le treillis de cube de données avec les hiérarchies de dimensions fournit un moyen d'agréger les données selon plusieurs niveaux de granularité qui fournissent un moyen intuitif pour les analystes afin de naviguer à différents niveaux de détail de l'information Casali et al. (2003). Toutefois, la hiérarchie introduit un problème fondamental de calcul de cube efficace : le nombre de cuboïdes dans un treillis augmente avec le nombre de dimensions ainsi que le nombre de niveaux de hiérarchies. Ainsi, le nombre total de chemins dépend du nombre des nœuds dans le treillis. Sachant que n_i est le nombre de niveaux des hiérarchies, selon Caron et Daniels (2008), le nombre de chemins dans un treillis est donné par la formule suivante.

$$nombre\ de\ chemins = \frac{(n_1 + n_2 + ... + n_k)!}{n_1!n_2!...n_k!}$$

Dans l'exemple de la Figure 2, nous avons 3 dimensions avec respectivement 1, 2 et 3 niveaux de hiérarchies, ce qui nous donne $\frac{(3+2+1)!}{3!*2!*1!} = \frac{720}{12} = 60$ chemins différents. Nous constatons alors qu'avec seulement 3 dimensions et 6 niveaux de hiérarchies, on obtient 60 chemins possibles dans le cube de données.

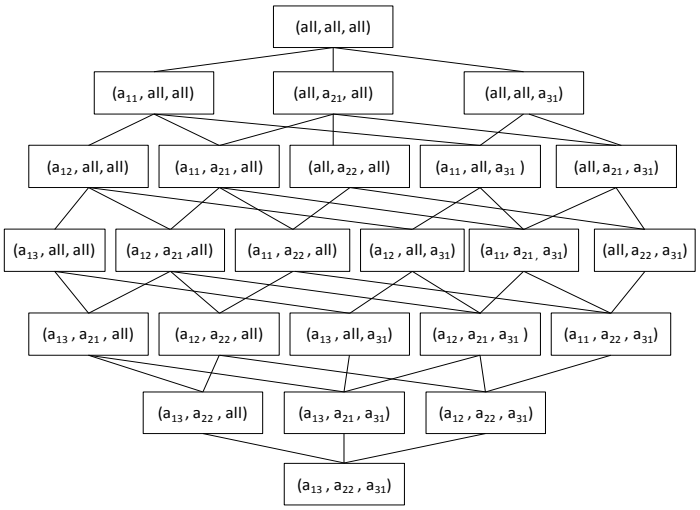

FIG. 2 – *Treillis combiné*

Définition 1. Ordre partiel dans un treillis de cuboïdes
L'arête entre deux cuboïdes c et c' représente la relation de dépendance entre ces deux cuboïdes. On dit que c est dépendant de c', noté $c \preceq c'$, si une requête répondue par c peut être aussi répondue par c', mais l'inverse n'est pas vrai. On dit aussi que $c \preceq c'$ si et seulement si c peut être calculé à partir de c'.

Notons que $\preceq$ impose un ordre partiel sur les nœuds et qu'il est transitif. Pour un ensemble d'éléments d'un treillis, deux éléments doivent avoir une borne supérieure et une borne inférieure selon l'ordre partiel $\preceq$. Cependant, dans la pratique, nous avons besoin uniquement des hypothèses suivantes : (a) $\preceq$ est un un ordre partiel, et (b) il existe un élément supérieur, un nœud duquel chaque nœud dépend.

Dans le treillis de la Figure 1, supposons que *Magasin*, *Produit* et *Date* correspondent respectivement à A_1, A_2 et A_3. On peut alors écrire $(all, \textit{Produit}, \textit{Date}) \preceq (all, \textit{Produit}, all)$. Par ailleurs, il existe des cuboïdes qui ne sont pas comparables entre eux en utilisant l'opérateur $\preceq$ comme par exemple les cuboïdes $(\textit{Magasin}, all, all)$ et $(all, \textit{Produit}, all)$.

Définition 2. Ancêtres et descendants
Étant donné un fait avec N dimensions, il existe 2^N cuboïdes. Nous définissons les ancêtres et les descendants d'un cuboïde c de la manière suivante :
$Anc(c) = \{c' \mid c \preceq c'\}$ et $Des(c) = \{c' \mid c' \preceq c\}$

Définition 3. Parents/Enfants
Le parent/enfant d'un nœud c dans un treillis est défini comme l'ancêtre/descendant immédiat de c. Le parent P (ou enfant E) de c peut être défini comme suit :
$P(c) = \{c' \mid c \preceq c', \nexists x, c \preceq x, x \preceq c'\}$ et $E(c) = \{c' \mid c' \preceq c, \nexists x, c' \preceq x, x \preceq c\}$
Par exemple dans la Figure 1, les ancêtres du cuboïde (A_1, A_2, all) sont les cuboïdes (A_1, all, all), (all, A_2, all) et (all, all, all) tandis que ses parents sont les cuboïdes (A_1, all, all) et (all, A_2, all).

Définition 4. Chemin de navigation dans un cube OLAP
Un chemin de navigation dans un cube de données est défini par une séquence ordonnée de requêtes $q_1, q_2, ..., q_n$, appliquées aux cuboïdes $c_1, c_2, ..., c_n$, respectivement, tels que $c_1 \diamond c_2, c_2 \diamond c_3, ..., c_{n-1} \diamond c_n$ où $\diamond$ est soit $\preceq$ soit $\succeq$. Autrement dit, étant donnée une requête, un utilisateur applique l'opérateur drill-down/roll-up pour passer d'un niveau supérieur à un niveau inférieur et vice-versa.

2.3 Modèle de Markov

Les chaînes de Markov (Markov (1971)) modélisent des relations entre des éléments dans le temps selon une hypothèse d'indépendance telle que la probabilité d'apparition d'un élément n'est dépendante que de l'élément précédent. Un modèle de Markov correspond à un simple graphe d'états, doté d'une fonction de transition probabiliste. Une chaîne de Markov désigne les processus qui commencent dans l'un de ces états, et évoluent d'état en état selon les transitions.

A chaque pas de temps, le modèle subit une transition qui va potentiellement modifier son état. Cette transition permet donc au système modélisé d'évoluer, selon une loi connue par avance. Néanmoins, cette loi de transition est probabiliste. En effet, l'évolution du système peut être incertaine, ou simplement mal connue. Cette fonction probabiliste permet donc d'exprimer simplement la loi d'évolution du modèle, sous la forme d'une matrice de probabilités. Cela ouvre donc la porte à un très grand nombre d'utilisations où l'évolution d'un système n'est connue qu'à travers des statistiques.

Un modèle de Markov peut être décrit de la manière suivante. Étant donné un ensemble d'états $S = \{s_1, s_2..., s_n\}$, le processus débute dans un de ces états et se propage successivement d'un état à l'autre. Chaque mouvement correspond à une étape du processus. Si le processus se trouve dans l'état s_i alors il peut se trouver dans l'état suivant s_j avec une probabilité p_{ij}. Cette probabilité ne dépend pas du chemin effectué depuis le début du processus mais uniquement de l'état s_i. L'ensemble des probabilités p_{ij}, $1 < i, j < n$, n étant le nombre total d'états composant le modèle, est représenté par la matrice de transitions notée MT. Notons également que le processus peut rester dans le même état d'une étape à l'autre avec une probabilité p_{ii}.

Pour définir totalement le modèle, il reste à définir un vecteur $u = (u_1, u_2, ...u_n)$ correspondant aux probabilités de débuter le processus dans l'un ou l'autre des états. La probabilité

de débuter dans l'état s_i est u_i. L'état s_i est appelé alors état initial.

Un modèle de Markov doit respecter les deux contraintes suivantes :
— La somme des probabilités des états initiaux est égale à 1 :$\sum_{i=1}^{n} u_i = 1$.
— La somme des probabilités des transitions partant d'un état est égal à 1 : $\forall i, 1 \leq i \leq n, \sum_{j=1}^{n} p_{ij} = 1$.

3 Recommandation collaborative de chemins de navigation

3.1 Principe

Afin de recommander des chemins de navigation pertinents dans un cube OLAP, nous utilisons l'historique des accès des utilisateurs extrait à partir des logs des sessions d'analyse. Étant donné qu'un cube de données peut être considéré comme un treillis de cuboïdes, notre approche tire profit de cette structure pour construire un modèle probabiliste dans lequel sont stockées des estimations de probabilités. Ces probabilités peuvent alors être utilisées pour calculer le chemin de navigation le plus probable. Nous utilisons le modèle de Markov pour modéliser le comportement de l'utilisateur lors de sa navigation dans un cube OLAP parce qu'il offre un moyen simple pour saisir la dépendance séquentielle entre un niveau d'analyse (état) à l'autre (nouvel état). En effet, l'utilisation du modèle de Markov pour la navigation a été étudiée et reconnue dans la littérature depuis plusieurs années pour prédire la requête de l'utilisateur dans le Web Stober et Nürnberger (2006), Eirinaki et al. (2005) et Zhu et al. (2002).

Dans ce cadre, notre approche consiste à prédire le ou les prochains cuboïdes qu'un utilisateur donné est susceptible de visiter ou consulter. Ainsi, il est possible d'effectuer de la recommandation à partir de ces prédictions, simplement en recommandant les navigations prédites ayant la plus grande probabilité.

Notre approche de recommandation de chemins de navigation se déroule en trois phases. (1) Tout d'abord, à partir d'un ensemble de sessions d'analyses issues des fichiers logs des utilisateurs, un prétraitement est nécessaire pour extraire les différentes sessions d'analyse. (2) Nous procédons ensuite à la construction du treillis d'accès (treillis de cuboïdes + probabilités de transition entre les nœuds en fonction de la fréquence des visites) appelé plus communément DAL (Data cube Access Lattice). (3) Enfin, nous exploitons le treillis d'accès pour recommander à l'utilisateur des chemins de navigation et l'aider à anticiper dans la navigation pour aller vers les faits les plus pertinents. Cette phase est elle-même effectuée en deux étapes : (i) construction de la liste des recommandations candidates en appliquant le modèle de Markov sur les sessions extraites, et (ii) ordonnancement de ces recommandations candidates. Ainsi, notre approche de recommandation de chemins de navigation fournit à l'utilisateur un ensemble de recommandations et, dans le même temps, met à jour le treillis d'accès.

3.2 Processus de recommandation de chemins de navigation

3.2.1 Étape 1 - Prétraitement

La première étape consiste à pétraiter le log des requêtes d'analyse. En effet, les tâches de prétraitement, y compris le nettoyage des données, l'identification de l'utilisateur et l'identifi-

cation des sessions peuvent être appliquées au log des sessions pour obtenir toutes les sessions d'analyse (des séquences de requêtes). Le log peut être divisé en sessions par de nombreuses façons telles que l'identificateur utilisateur (*user id*), la date (*timestamp*) et la durée d'une session. Nous combinons le nom d'utilisateur et le seuil de délai d'attente (*timeout treshold*) pour délimiter les sessions d'analyse. Le seuil de délai d'attente est fixé à moins de 12 heures dans le système SQL Server. Ainsi, le log des sessions est divisé en plusieurs parties, tout d'abord selon l'identificateur de l'utilisateur, puis nous comparons le *timestamp* de deux requêtes consécutives. Si le délai d'attente entre ces deux requêtes dépasse 12 heures, la requête ayant le plus petit *timestamp* représente un point de départ pour une nouvelle session d'analyse.

Session	**Fréquence**
$s_1 \rightarrow s_2 \rightarrow s_4 \rightarrow s_7$	3
$s_1 \rightarrow s_2 \rightarrow s_4 \rightarrow s_5 \rightarrow s_7$	2
$s_1 \rightarrow s_3 \rightarrow s_4 \rightarrow s_7$	4
$s_1 \rightarrow s_2 \rightarrow s_4 \rightarrow s_6 \rightarrow s_7$	2
$s_1 \rightarrow s_4 \rightarrow s_7$	2
$s_1 \rightarrow s_4 \rightarrow s_5 \rightarrow s_7$	1
$s_1 \rightarrow s_4 \rightarrow s_6 \rightarrow s_7$	2
$s_1 \rightarrow s_2 \rightarrow s_5 \rightarrow s_7$	3
$s_5 \rightarrow s_4 \rightarrow s_2$	1
$s_6 \rightarrow s_4 \rightarrow s_3$	1
$s_4 \rightarrow s_2$	2

TAB. 1 – *Sessions d'analyse identifiées dans le log*

Le Tableau 1 présente un exemple d'identification de différentes sessions d'analyse dans un log. On trouve dans la première colonne une collection de sessions de navigation avec l'état de départ et l'état final. La deuxième colonne du Tableau 1 nous donne la fréquence d'une session d'analyse qui représente le nombre de fois où la séquence de requête correspondante est traversée ou visitée dans le log.

3.2.2 Étape 2 - Construction du treillis d'accès

La deuxième étape consiste à construire le treillis d'accès qui est le treillis de cuboïdes avec les probabilités de transition entre les nœuds en fonction de l'historique de navigation (sessions et fréquence des visites). Une fois, les différentes sessions d'analyse obtenues, elles peuvent correspondre à un graphe pondéré appelé modèle de Markov. Le modèle de Markov se compose d'un ensemble d'états qui représentent les requêtes des utilisateurs où deux états successifs sont reliés par un lien ou une arête. Chaque état ou cuboïde est défini par une identité ($s_1, s_2, etc.$). Chaque lien ou arête est désigné par un certain nombre de visites des séquences de requêtes. La chaîne de Markov est définie par un ensemble d'états avec leurs probabilités initiales et une matrice de transition MT. L'ensemble des états est composé de l'état initial, l'état final et les états intermédiaires qui correspondent aux cuboïdes visités. Dans ce contexte, nous proposons de modéliser le cube de données comme une chaîne de Markov où les cuboïdes correspondent aux états et le chemin de navigation correspond aux transitions entre les états. Les chemins de la chaîne ayant la probabilité la plus élevée seront les chemins préférés dans

le cube OLAP. Pour cela, nous calculons les différentes probabilités en se basant sur le modèle de Markov. Une fois le treillis d'accès construit, nous pouvons l'exploiter pour recommander des chemins de navigation à l'utilisateur. Étant donnée une requête courante de l'utilisateur, nous identifions à quel état elle correspond dans le modèle de Markov. À partir de cet état et en utilisant la matrice de transitions, nous pouvons calculer plusieurs probabilités de passage de l'état présent à un autre état. Un exemple complet de construction du treillis d'accès est présenté à la section 4.

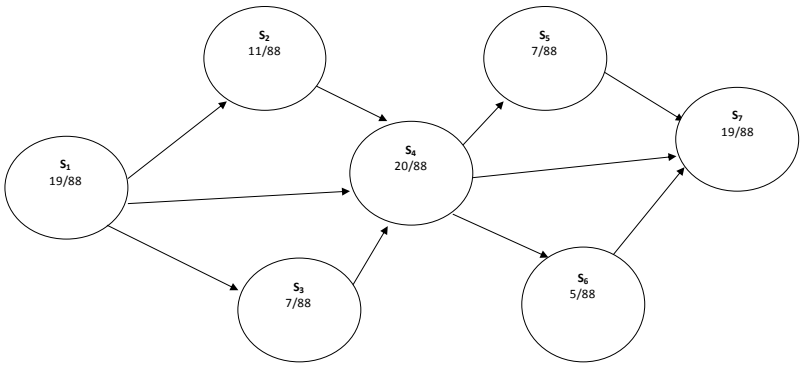

FIG. 3 – *Modèle de Markov des sessions d'analyse*

La Figure 3 représente le modèle de sessions données dans le Tableau 1. Chaque état a une identité et une probabilité d'état.

Probabilité d'état. La probabilité d'état P(s) (*state probability*) représente la probabilité de soumettre une requête dans un cube OLAP (cela correspond à un cuboïde) est obtenue par la formule suivante : $P(s) = \frac{N_{s_i}}{N}$ où :

— N_{s_i} : nombre de fois où la requête s_i a été soumise.

— N : nombre total de requêtes dans le log des utilisateurs.

Par exemple, à partir du Tableau 1, nous pouvons calculer la probabilité de l'état s_2. Celle-ci est égale à $11/88$ car la requête s_2 a été visitée 11 fois et que le nombre total de requêtes est 88.

Probabilité de transition. Chaque lien entre les nœuds d'un treillis représente la probabilité de transition (TP) qui est calculée en fonction du nombre de fois où le lien correspondant est suivi ou visité et du nombre de fois que le nœud d'ancrage a été visité. La probabilité de transition est représentée dans la matrice de transitions MT qui enregistre les probabilités de transition. Ainsi, la probabilité de transition peut être calculée par la formule suivante :

$$TP(s_i \rightarrow s_j) = \frac{N_{(s_i \rightarrow s_j)}}{N_{s_i}}$$

où $N(s_i \rightarrow s_j)$ est le nombre de fois où le lien entre s_i et s_j a été emprunté.

Par conséquent, notre treillis d'accès est représenté par la matrice de transitions MT qui peuvent être calculées à partir des probabilités d'états. A partir de cette matrice MT et la

position courante (état actuel), nous allons calculer les probabilités de tous les chemins de navigation possibles dans le graphe de Markov commençant par cet état courant. Par exemple, nous pouvons calculer la probabilité de transition de s_2 à s_4 comme suit : $TP(s_2 \rightarrow s_4) = 5/11$ et la probabilité de transition de s_4 à s_6 comme suit : $TP(s_4 \rightarrow s_6) = 4/20$. De plus, nous pouvons calculer la probabilité de transition dans le sens contraire comme par exemple la probabilité de transition de s_4 à s_2 qui est égale à $TP(s_4 \rightarrow s_2) = 3/20$. Le Tableau 2 montre la matrice de transitions MT qui représente toutes les probabilités de transition possibles.

MT	s_1	s_2	s_3	s_4	s_5	s_6	s_7
s_1	0	8/19	6/19	5/19	0	0	0
s_2	3/11	0	0	5/11	3/11	0	0
s_3	0	0	0	6/7	1/7	0	0
s_4	0	3/20	1/20	0	3/20	4/20	9/20
s_5	0	0	0	1/7	0	0	6/7
s_6	0	0	0	1/5	0	0	4/5
s_7	0	0	0	0	0	0	0

TAB. 2 – *Matrice de transitions*

Probabilité de chemin. Un chemin est une séquence finie d'états qui sont accessibles dans l'ordre de leur parcours dans le cube de données sous-jacent. Dans ce contexte, la probabilité d'un chemin de navigation est estimée par le produit de la probabilité initiale du premier état du chemin de navigation et les probabilités des transitions. La règle de la chaîne de Markov est appliquée afin de calculer toutes les probabilités de chemins. Ainsi, la probabilité d'un chemin de navigation PP peut être calculée par la formule suivante :

$$PP(s_i \rightarrow s_j) = P(s_i \prod TP(s_i \rightarrow s_j))$$

Par exemple, la probabilité estimée du chemin $(s_1 \rightarrow s_3 \rightarrow s_4 \rightarrow s_7)$ est égale à : $19/88 * 6/19 * 6/7 * 9/20$. Le Tableau 3 montre les probabilités de chemins des sessions données dans le log comme des exemples de calcul de probabilités de chemins.

3.2.3 Étape 3 - Recommandation collaborative de chemins de navigation

Les systèmes de recommandation sont généralement classés selon deux catégories : des systèmes basés sur le contenu et ceux basés sur le filtrage collaboratif Adomavicius et Tuzhilin (2005). Les systèmes basés sur le contenu recommandent à l'utilisateur des éléments similaires à ceux qui l'ont intéressé dans le passé, tandis que les systèmes basés sur le filtrage collaboratif recommandent des éléments qui ont intéressé des utilisateurs qui lui sont similaires. Ce travail s'inscrit dans une approche de recommandation collaborative impliquant plusieurs utilisateurs parce que nous exploitons les sessions d'analyse de l'ensemble des utilisateurs du système décisionnel. L'idée d'exploiter ce que les autres utilisateurs ont fait pour produire des recommandations est très populaire dans le domaine de la recherche d'informations Adomavicius et Tuzhilin (2005), et dans l'exploitation des usages du Web (Web Usage Mining) Srivastava et al. (2000). Notre contribution est d'adapter ces techniques existantes à l'OLAP.

Généralement, dans la recommandation collaborative, le problème de base est le suivant : on dispose d'un ensemble de m éléments (livres, films, objets, etc.), de n utilisateurs et d'une

Session	**Fréquence**	**Probabilité de chemin**
$s_1 \rightarrow s_2 \rightarrow s_4 \rightarrow s_7$	3	19/88 * 8/19 * 5/11 * 9/20
$s_1 \rightarrow s_2 \rightarrow s_4 \rightarrow s_5 \rightarrow s_7$	2	19/88 * 8/19 * 5/11 * 3/20 * 6/7
$s_1 \rightarrow s_3 \rightarrow s_4 \rightarrow s_7$	4	19/88 * 6/19 * 6/7 * 9/20
$s_1 \rightarrow s_2 \rightarrow s_4 \rightarrow s_6 \rightarrow s_7$	2	19/88 * 6/19 * 6/7 * 4/20 * 4/5
$s_1 \rightarrow s_4 \rightarrow s_7$	2	19/88 * 5/19 * 9/20
$s_1 \rightarrow s_4 \rightarrow s_5 \rightarrow s_7$	1	19/88 * 5/19 * 4/20 * 4/5
$s_1 \rightarrow s_4 \rightarrow s_6 \rightarrow s_7$	2	19/88 * 5/19 * 4/20 * 4/5
$s_1 \rightarrow s_2 \rightarrow s_5 \rightarrow s_7$	3	7/88 * 1/7 * 3/20
$s_5 \rightarrow s_4 \rightarrow s_2$	1	5/88 * 1/5 * 1/20
$s_6 \rightarrow s_4 \rightarrow s_3$	1	20/88 * 3/20
$s_4 \rightarrow s_2$	2	3/20

TAB. 3 – *Probabilités de chemins de navigation*

matrice d'utilité $R = (r_{ij}, i = 1, ...n, j = 1, ...m)$ (*utility matrix* ou bien *rating matrix*) telle que :

— $r_{ij} \in R$ signifie que l'utilisateur i a attribué la note r_{ij} à l'élément j
— $r_{ij} = *$ signifie que la note attribuée par l'utilisateur i à l'élément j n'est pas connue

Dans le cadre de nos travaux sur la recommandation de chemins de navigation, la matrice d'utilité est la matrice de transitions MT qui représente le treillis d'accès. Par conséquent, à partir de l'état actuel de l'utilisateur dans le treillis d'accès et la matrice de transitions MT, nous recommandons à l'utilisateur les chemins de navigation les plus pertinents à emprunter. Nos recommandations sont obtenues à partir de la matrice de transitions. Ceci peut être expliqué par l'algorithme de recommandation de chemins de navigation (Algorithme 1) qui calcule des recommandations candidates en se basant sur la requête courante et l'historique de l'utilisateur représenté par la matrice de transitions.

Les notations utilisées dans l'algorithme sont les suivantes :

— Ch désigne la file d'attente des nœuds (cuboïdes dans le treillis).
— c est le premier nœud de Ch.
— C est l'ensemble des cuboïdes dans le treillis.
— NC est le nœud courant qui représente le début d'une session d'analyse.
— $TP[i][j]$ désigne la probabilité d'une arête reliant les nœds i et j dans le treillis de cuboïdes. C'est la probabilité que la prochaine requête sera adressée au cuboïde j sachant que la requête courante est adressée au cuboïde i. Le log de sessions d'analyse est utilisé pour déterminer les probabilités des arêtes. Ces valeurs sont mises à jour entre deux sessions successives.
— CA désigne la condition d'arrêt fixée par l'utilisateur. Elle peut être présentée par une longueur de chemin ou une valeur de seuil de probabilité exigées par l'utilisateur.

Algorithme 1 : NAPARE (MT, NC, CA)

Entrées :
MT : Matrice de transitions
NC : Nœud courant
CA : Condition d'arrêt
Sortie :
Ch : Chemin de navigation à recommander
Début
  ajouter NC au Ch ;
  Tant que ¬ *vide(*Ch*)* **faire**
    c = premier élément de Ch ;
    Pour chaque a *dans* $Parent(NC) \cap C$ *ou* a *dans* $Enfant(NC) \cap C$ **faire**
      Si *(*¬ CA*)* **alors**
        calculer Max (TP[NC][a]) ;
        ajouter a à la fin de Ch ;
      Finsi
    Finpour
  fintantque
  Retourner *Chemin de navigation* Ch ;
  `// Retourner le chemin de navigation le plus pertinent`
Fin

Notre système de recommandation NAPARE utilise comme base le treillis d'accès et la requête courante de l'utilisateur. Tout d'abord, le modèle de Markov est construit à partir du treillis de cuboïdes et l'historique des requêtes des utilisateurs. Le résultat obtenu est le treillis d'accès (DAL) qui permet de résumer le comportement interrogatoire des utilisateurs passés. Il est représenté par la matrice de transitions (MT).

L'algorithme commence avec la première requête de l'utilisateur qui représente le nœud courant ou l'état actuel. L'algorithme effectue une première recherche étendue pour calculer la probabilité de chaque nœud. La condition d'arrêt peut être la longueur du chemin ou une valeur de seuil appliquée à la probabilité de l'arête. Par exemple, si la probabilité d'un nœud est inférieure à la valeur du seuil ou si la longueur du chemin de navigation dépasse une certaine valeur, alors on pourrait arrêter la poursuite de nœuds le long de ce chemin. Ensuite, NAPARE cherche des correspondances entre la session active de l'utilisateur et les états du modèle (différents nœuds du treillis d'accès). Enfin, le(s) chemin(s) qui a(ont) la plus grande probalilité sera(seront) recommandé(s) à l'utilisateur.

4 Application de NAPARE sur un cube OLAP

Prenons l'entrepôt de données *Foodmart* comme exemple pour y appliquer notre système de recommandation NAPARE. Nous considérons le cube de données *Sales* qui calcule le nombre de ventes (mesure : *Store Sales*) par les dimensions magasin (*Store*), date (*Time*) et produit (*Product*). La dimension *Time* possède trois niveaux de hiérarchies *Time*, *Month* et *Year*, la dimension *Store* est hiérarchisée selon deux niveaux *Store* et *City*. Le treillis de cuboïdes avec les hiérarchies de dimensions est donné dans la Figure 4 (treillis d'accès sans les

probalibilités). C'est un treillis qui comporte 24 nœuds qui représentent tous les cuboïdes du cube OLAP. Le treillis d'accès (Figure 4) est présenté par une matrice de transitions qui est égale à la matrice d'adjacence du treillis avec les probabilités de transitions entre les arêtes. C'est une matrice carrée symétrique (graphe non orienté) d'ordre 24 (nombre de nœuds du treillis de cuboïdes).

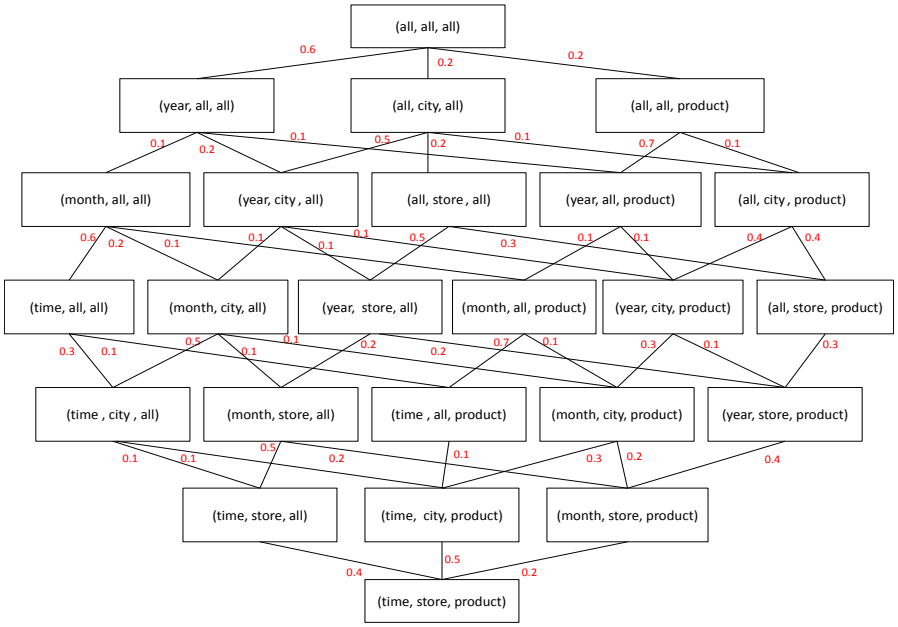

FIG. 4 – *Treillis d'accès*

La probabilité qu'un utilisateur visite un cuboïde c' sachant qu'il interroge un cuboïde c est égale à la probabilité de transition entre les deux nœuds dans la matrice de probabilité du $n^{\text{ième}}$ degré où n est égal au nombre de sauts (longueur du chemin) pour atteindre l'état (cuboïde) c' à partir de l'état (cuboïde) c. Supposons qu'un utilisateur commence son analyse à partir du nœud NC (all, all, all) qui correspond à sa première requête. Nous appliquons l'algorithme NAPARE (Algorithme 1) sur la matrice de transitions MT présentée dans le Tableau 4 de la manière suivante : NAPARE (MT, (all, all, all), n=3) avec la condition d'arrêt n=3 qui est la longueur du chemin fixée par l'utilisateur. Par conséquent, les chemins de navigation les plus pertinents sont les chemins qui ont la probabilité maximale. Dans notre cas, nous obtenons 3 chemins candidats car ils ont tous la même probabilité maximale égale à $0.6{*}0.2{*}0.1 = 0.072$.

- Chemin 1 : $(all, all, all) \rightarrow (year, all, all) \rightarrow (year, city, all) \rightarrow (month, city, all)$
- Chemin 2 : $(all, all, all) \rightarrow (year, all, all) \rightarrow (year, city, all) \rightarrow (year, store, all)$
- Chemin 3 : $(all, all, all) \rightarrow (year, all, all) \rightarrow (year, city, all) \rightarrow (year, city, product)$

Si l'utilisateur ne choisit aucun des chemins candidats qui lui sont proposés et accède au cuboïde $(all, all, product)$, NAPARE calcule de nouveau les probalilités des chemins partant du cuboïde courant (chemin 1 et chemin 2) et recommande ensuite à l'utilisateur le chemin 1 puisqu'il a la plus grande probabilité.
- Chemin 1 :
$(all, all, product) \rightarrow (year, all, product) \rightarrow (month, all, product) \rightarrow (time, all, product)$ ayant la probabilité $0.7 * 0.1 * 0.7 = 0.049$.
- Chemin 2 :
$(all, all, product) \rightarrow (year, all, product) \rightarrow (year, city, product) \rightarrow (month, city, product)$ ayant la probabilité $0.7 * 0.1 * 0.3 = 0.021$.

MT	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	0	0.6	0.2	0.2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0.6	0	0	0	0.1	0.2	0	0.1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0.2	0	0	0	0	0.5	0.2	0	0.1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0.2	0	0	0	0	0	0	0.7	0.1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0.1	0	0	0	0	0	0	0	0.6	0.2	0	0.1	0	0	0	0	0	0	0	0	0	0	0
6	0	0.2	0.5	0	0	0	0	0	0	0	0.1	0.1	0	0.1	0	0	0	0	0	0	0	0	0	0
7	0	0	0.2	0	0	0	0	0	0	0	0	0.5	0	0	0.3	0	0	0	0	0	0	0	0	0
8	0	0.1	0	0.7	0	0	0	0	0	0	0	0	0.1	0.1	0	0	0	0	0	0	0	0	0	0
9	0	0	0.1	0.1	0	0	0	0	0	0	0	0	0	0.4	0.4	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0.6	0	0	0	0	0	0	0	0	0	0	0.3	0	0.1	0	0	0	0	0	0
11	0	0	0	0	0.2	0.1	0	0	0	0	0	0	0	0	0	0.5	0.1	0	0.1	0	0	0	0	0
12	0	0	0	0	0	0.1	0.5	0	0	0	0	0	0	0	0	0	0.2	0	0	0.2	0	0	0	0
13	0	0	0	0	0.1	0	0	0.1	0	0	0	0	0	0	0	0	0	0.7	0.1	0	0	0	0	0
14	0	0	0	0	0	0.1	0	0.1	0.4	0	0	0	0	0	0	0	0	0	0.3	0.1	0	0	0	0
15	0	0	0	0	0	0	0.3	0	0.4	0	0	0	0	0	0	0	0	0	0	0.3	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0.3	0.5	0	0	0	0	0	0	0	0	0	0.1	0.1	0	0
17	0	0	0	0	0	0	0	0	0	0	0.1	0.2	0	0	0	0	0	0	0	0	0.5	0	0.2	0
18	0	0	0	0	0	0	0	0	0	0.1	0	0	0.7	0	0	0	0	0	0	0	0	0.1	0	0
19	0	0	0	0	0	0	0	0	0	0	0.1	0	0.1	0.3	0	0	0	0	0	0	0	0.3	0.2	0
20	0	0	0	0	0	0	0	0	0	0	0	0.2	0	0.1	0.3	0	0	0	0	0	0	0	0.4	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.1	0.5	0	0	0	0	0	0	0.4
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.1	0	0.1	0.3	0	0	0	0	0.5
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.2	0	0.2	0.4	0	0	0	0.2
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.4	0.5	0.2	0

TAB. 4 – *Matrice de transitions représentative du treillis d'accès*

5 Développement et validation

Nous avons développé un prototype en Java de notre système NAPARE pour mettre en oeuvre notre méthode de recommandation de chemins de navigation dans un cube OLAP. Nous avons appliqué notre méthode sur un cube de données "Sales" obtenu à partir de l'entrepôt de données *Foodmart*. NAPARE permet de simuler le déplacement d'un utilisateur dans le cube de données. Le principe est simple : on place l'utilisateur dans un nœud initial (correspondant à sa première requête), puis on utilise les distributions de probabilités de transition pour décider à chaque pas de temps où il se rend. L'idée est alors de garder le nombre de fois où l'utilisateur passe dans chaque nœud, ce qui permettrait de construire une distribution de probabilités (si convergence). Le principe de base est d'attribuer à chaque nœud une valeur (ou un score) proportionnelle au nombre de fois que passerait par ce nœud un utilisateur parcourant le treillis de cuboïdes en cliquant aléatoirement, sur un des liens apparaissant sur chaque cuboïde. Le déplacement de l'utilisateur peut alors être assimilé à une marche aléatoire sur le graphe de cube de données, autrement dit il s'agit du processus de Markov. En effet, à chaque fois que l'utilisateur choisit le cuboïde (nœud dans le treillis) suivant pour poursuivre sa navigation, le choix de ce nœud dépend des nœuds précédemment visités.

Nous avons testé notre prototype sur une charge de 40 sessions d'analyse relatives à l'entrepôt de données *Foodmart*. Ces sessions d'analyse comptent 120 requêtes. Nous avons ensuite évalué la qualité de nos recommandations en utilisant la mesure statistique de précision MAE (Mean Absolute Error) (Shardanand et Maes (1995)) qui consiste à évaluer la différence existante entre les chemins prédits et les chemins réellement empruntés par les utilisateurs. La MAE calcule, pour chaque paire <chemin-prédiction>, la moyenne d'erreur absolue entre les chemins prédits $Pred(u_a, i)$ et les chemins réellement empruntés par les utilisateurs $v(u_a, i)$ (n représente le nombre total de chemins prédits). Plus la valeur de MAE est faible, plus les prédictions sont précises et plus le système de recommandation est performant.

$$MAE = \frac{\sum_{i=1}^{n} |v(u_a, i) - Pred(u_a, i)|}{n}$$

Par ailleurs, les chemins de navigation recommandés par notre système NAPARE sont de bonne qualité puisque nous utilisons le treillis de cuboïdes qui nous garantit d'avoir toujours des chemins valides.

6 Conclusion

Dans cet article nous avons présenté NAPARE, un système de recommandation collaborative de chemins de navigation dans un cube OLAP. Notre système tient compte des précédentes navigations réalisées par l'ensemble des utilisateurs du cube, partitionne le log des requêtes qui peut être volumineux, tient compte des hiérarchies dans le treillis des cuboïdes et utilise le modèle de Markov pour prédire et recommander la session d'analyse suivante à l'utilisateur. Le calcul des chemins de navigation les plus pertinents, étant donnée une session d'analyse courante, se fait en temps réel au fur et à mesure du processus de navigation de l'utilisateur. Par ailleurs, les expérimentations que nous avons menées sur le benchmark *Foodmart* montrent que les recommandations qui sont calculées par notre système NAPARE sont de bonne qualité.

Une perspective directe de ce travail est de procéder à l'élagage du treillis de cuboïdes étant donnée que la taille du cube de données peut être conséquente. En effet, le processus d'élagage consisterait à éliminer les nœuds inintéressants du treillis d'accès (nœuds inexistants dans le log de requêtes) que les utilisateurs ne visitent pas afin d'alléger leurs tâches de navigation. Par ailleurs, nous souhaitons intégrer le profil utilisateur au sein de NAPARE afin d'améliorer la qualité de ses recommandations.

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. *Journal of IEEE Trans. Knowl. Data Eng. Volume 17*(6), pages 734–749.

Bentayeb, F., O. Boussaïd, C. Favre, F. Ravat, et O. Teste (2009). Personnalisation dans les entrepôts de données : bilan et perspectives. In *Actes des 5èmes journées francophones sur les Entrepôts de Données et l'Analyse en ligne, EDA 2009, Montpellier, France, Juin 4-5, 2009*, pp. 7–22.

Caron, E. et H. Daniels (2008). Explanation of exceptional values in multi-dimensional business databases. *European Journal of Operational Research Volume 188*(3), pages 884 – 897.

Casali, A., R. Cicchetti, et L. Lakhal (2003). Cube lattices : A framework for multidimensional data mining. In *SDM*, pp. 304–308.

Dittrich, J.-P., D. Kossmann, et A. Kreutz (2005). Bridging the gap between olap and sql. In *VLDB*, Trondheim, Norway, pp. 1031–1042.

Eirinaki, M., M. Vazirgiannis, et D. Kapogiannis (2005). Web path recommendations based on page ranking and markov models. In *WIDM*, Bremen, Germany, pp. 2–9.

Gray, J., A. Bosworth, A. Layman, et H. Pirahesh (1996). Data cube : A relational aggregation operator generalizing group-by, cross-tab, and sub-total. In *ICDE*, New Orleans, Louisiana USA, pp. 152–159.

Harinarayan, V., A. Rajaraman, et J. D. Ullman (1996). Implementing data cubes efficiently. In *International Conference on Management of Data*, Montreal, Canada, pp. 205–216.

Khemiri, R. et F. Bentayeb (2013). FIMIOQR : Frequent itemsets mining for interactive olap query recommendation. In *The Fifth International Conference on Advances in Databases, Knowledge, and Data Applications DBKDA 2013, Seville, Spain, January 27 - February 1, 2013*, pp. 9–14.

Markov, A. (1971). Extension of the limit theorems of probability theory to a sum of variables connected in a chain. In R. Howard (Ed.), *Dynamic Probabilistic Systems (Volume I : Markov Models)*, Chapter Appendix B, pp. 552–577. New York City : John Wiley & Sons, Inc.

Sarawagi, S. (2000). User-adaptive exploration of multidimensional data. In *VLDB*, pp. 307–316.

Shardanand, U. et P. Maes (1995). Social information filtering : Algorithms for automating "word of mouth". In *CHI*, Denver, Colorado USA, pp. 210–217.

Srivastava, J., R. Cooley, M. Deshpande, et P.-N. Tan (2000). Web usage mining : Discovery and applications of usage patterns from web data. *SIGKDD Explorations 1*(2), 12–23.

Stober, S. et A. Nürnberger (2006). DAWN - A system for context-based link recommendation in web navigation. In *KES*, Bournemouth, UK, pp. 763–770.

Zhu, J., J. Hong, et J. G. Hughes (2002). Using markov chains for link prediction in adaptive web sites. In *Soft-Ware*, Belfast, Northern Ireland, pp. 60–73.

Summary

This paper describes NAPARE, a NAvigation PAth REcommendation system for OLAP users. NAPARE is based on users log file analysis sessions and Markov chains that predict the next analysis query from the only current one.