Résumé de cubes de données multidimensionnelles à l'aide de règles floues[1]

Yeow Wei Choong*/***, Anne Laurent**
Dominique Laurent***, Pierre Maussion*

*HELP Institute - BZ-2 Pusat Bandar Damansara - 50490 Kuala Lumpur - MALAYSIA
choongyw@help.edu.my,maussp@help.edu.my
**LIRMM - Université Montpellier II - 161 rue Ada - 34392 Montpellier Cedex 5
anne.laurent@lirmm.fr
***LICP - Université de Cergy-Pontoise - 2, av Chauvin - 95302 Cergy-Pontoise
dominique.laurent@dept-info.u-cergy.fr

Résumé. Dans le contexte des entrepôts de données, et des magasins de données multidimensionnelles, les outils OLAP fournissent des moyens aux utilisateurs de naviguer dans leurs données afin d'y découvrir des informations pertinentes. Cependant, les données à traiter sont souvent très volumineuses et ne permettent pas une exploration systématique et exhaustive. Il s'agit donc de développer des traitements automatisés facilitant la visualisation et la navigation dans les données. Dans cet article, nous étudions une méthode originale permettant de construire et d'identifier de manière automatique et efficace des blocs de données similaires présents dans les cubes de données pouvant être exprimés sous la forme de règles. Cette méthode est fondée sur l'utilisation combinée d'un algorithme par niveaux (de type Apriori) et de la théorie des sous-ensembles flous. Cette théorie nous permet en effet de pallier les problèmes posés par le fait que les blocs de données calculés par notre algorithme peuvent se recouvrir.

MOTS CLES : Bases de données multidimensionnelles, algorithmes par niveaux, règles floues, visualisation de données.

1 Introduction

Les bases de données multidimensionnelles sont étudiées et utilisées depuis maintenant un peu plus de dix ans [Codd *et al.*, 1993]. Les bases de données relationnelles, définies pour des traitements de type OLTP (*On-Line Transactional Processing*) pour la production de données, se sont en effet révélées insatisfaisantes pour analyser et exploiter les gros volumes de données produits dans le cadre des traitements OLAP (*On-Line Analytical Processing*). Les bases de données multidimensionnelles ont alors été proposées à cette fin.

Les données, regroupées dans des entrepôts de données (ou *Data Warehouses*), sont sélectionnées pour construire des magasins de données (ou *Data Marts*) qui regroupent des données dédiées à une tâche d'analyse particulière. Ces magasins de données sont

[1]Ces travaux ont été partiellement financés par l'Ambassade de France en Malaisie.

le plus souvent modélisés sous la forme d'une base de données multidimensionnelles. Dans ce contexte, les données sont modélisées sous la forme de tableaux multidimensionnels appelés cubes de données définis en fonction de plusieurs dimensions. La donnée présente à l'intérieur du cube est appelée la *mesure*. Par exemple, le tableau suivant illustre un cube de données réduit à deux dimensions $PRODUIT$, $VILLE$ dont la mesure représente le nombre de ventes :

PRODUIT

P1	6	6	8	5	5	2	
P2	6	8	5	5	6	75	
P3	8	5	5	2	2	8	
P4	8	8	8	2	2	2	
	V1	V2	V3	V4	V5	V6	**VILLE**

Les outils OLAP fournissent des moyens aux utilisateurs de naviguer dans leurs données afin d'y découvrir des informations pertinentes. Plusieurs opérations de navigation et de manipulation des cubes de données sont définies comme par exemple la rotation (pour changer de point de vue sur les données en pivotant le cube et donc les dimensions considérées), la sélection, *etc.*

Cependant, les données à traiter sont souvent très volumineuses et ne permettent pas une exploration systématique et exhaustive. De plus, les utilisateurs de tels systèmes, experts des données, ne sont pas informaticiens et ne disposent donc pas des connaissances nécessaires pour poser des requêtes et bénéficier de leurs résultats. Si des outils de *reporting* ou d'analyse existent, il reste toujours difficile de visualiser les données pour en dégager des informations pertinentes. Typiquement, le nombre de dimensions varie selon les cubes construits, mais il est presque toujours supérieur à 4, ce qui rend difficile la visualisation de telles données.

Dans cet article, nous proposons des traitements automatisés facilitant la visualisation et la navigation dans les données. Nous étudions une méthode originale permettant de construire et d'identifier de manière automatique et efficace des blocs de données similaires présents dans les cubes de données. Chaque bloc est en fait un sous-ensemble des données prenant la forme d'un sous-hypercube, les blocs irréguliers n'étant pas autorisés. Sur l'exemple précédent, le sous-ensemble de données correspondant aux produits $P1$, $P2$ et à la ville $V1$ constitue un bloc de valeur de mesure homogène (6). Chaque bloc de données est exprimé sous la forme d'une règle pour en faciliter l'analyse. Par exemple, la règle associée au bloc précédemment présenté est :

Si PRODUIT = $P1$ **ou** $P2$ **et** VILLE = $V1$ **Alors** Ventes = 6.

Si dans ce cas la valeur de mesure est la même pour toutes les cellules du bloc, cela n'est pas toujours le cas. Par exemple, il existe un bloc associé à la valeur de mesure 5 correspondant aux produits $P1$, $P2$, $P3$ et aux villes $V3$ et $V4$ qui ne contient pas uniquement la valeur 5. Ce bloc est néanmoins considéré comme intéressant puisque *la plupart* des cellules qui le composent contiennent la même valeur. De même, il existe un bloc de valeurs 2 pour la zone correspondant aux produits $P3$, $P4$ et aux villes $V4$, $V5$, $V6$. Ces deux blocs se recouvrent puisqu'ils ont en commun la cellule correspondant au produit $P3$ et à la ville $V4$.

Il se produit donc des cas de recouvrement entre les blocs découverts, qu'il s'agit de traduire lors de la génération des règles. Pour ce faire, nous utilisons la théorie des sous-ensembles flous. Ce formalisme nous permet de représenter des informations du

type : pour le produit $P2$ et *dans une moindre mesure* pour le produit $P3$.

La méthode proposée dans cet article est donc fondée sur l'utilisation combinée des algorithmes par niveaux (fondés sur l'algorithme APriori) et de la théorie des sous-ensembles flous. L'utilisation de tels algorithmes est rendue nécessaire par la volonté de proposer des méthodes efficaces passant à l'échelle. Le but de notre travail est d'identifier le plus rapidement possible les blocs de données représentés sur la Fig. 1, d'en définir les recouvrements, et d'y associer des règles, floues ou non.

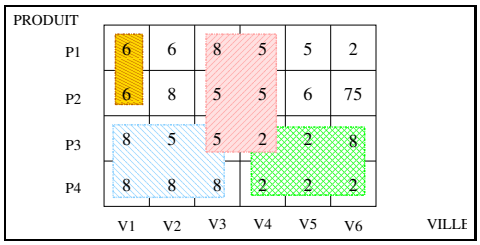

FIG. 1 – Exemple d'un cube et des blocs associés

2 Bases de données multidimensionnelles

Les masses de données stockées dans les entrepôts de données sont des *collections de données orientées sujet, intégrées, non volatiles, historisées et résumées* [Inmon, 1992]. Destinés à l'interrogation et à l'aide à la décision, les systèmes reposant sur cette technologie sont différents des bases de données de production classiques (par exemple pour la gestion du stock d'un magasin). On distingue les outils OLAP (On-Line Analytical Processing) destinés à l'analyse des outils OLTP (On-Line Transactional Processing) destinés à la production de données. Le modèle relationnel, parfaitement adapté aux systèmes transactionnels, s'est révélé inadapté dans le contexte OLAP. Les bases de données multidimensionnelles ont donc été proposées [Codd *et al.*, 1993]. Les données sont alors considérées sous la forme de tableaux multidimensionnels.

Il n'existe pas à l'heure actuelle de définition consensuelle concernant la représentation et la manipulation des bases de données multidimensionnelles. De manière générale, on appelle *base de données multidimensionnelle* un ensemble d'*hypercubes* (ou plus simplement *cubes*). Chaque cube est défini à partir de *dimensions*. La *mesure* est contenue dans les cellules du cube. Par exemple, le cube que nous décrivons dans cet article correspond au cube des ventes (qui constitue la mesure) défini à partir des dimensions PRODUIT et VILLE. Les dimensions peuvent être pourvues de *hiérarchies,* par exemple pour décrire les données au niveau des départements et non plus des villes.

Des opérations sont définies afin de manipuler ces cubes, comme par exemple la sélection, la projection, la rotation, l'inversion de l'ordre des valeurs de dimension. Cette dernière opération, nommée *switch* en anglais, permet, sans altérer les données présentes dans le cube, d'en modifier la représentation. Par exemple, le cube du tableau présenté précédemment contient les mêmes données que le cube suivant, mais est représenté de manière différente, les valeurs des dimensions ayant été inversées :

PRODUIT

P4	8	8	8	2	2	2	
P2	5	6	8	5	6	75	
P1	8	6	6	5	5	2	
P3	5	8	5	2	2	8	
	V3	V1	V2	V4	V5	V6	**VILLE**

On trouvera dans [Choong *et al.*, 2003] une étude détaillée des représentations pour un cube donné. Le but de cet article est d'aider à la visualisation des cubes de données en permettant la construction automatique de blocs de données. Dans cet article, nous considérons un cube à k dimensions C fixé et une de ses représentations, également fixée. On appelle alors bloc de données un sous-ensemble de cellules du cube formant un sous-cube :

Définition 1 - Bloc de données. *Un bloc de données b est un ensemble de cellules défini sur un cube C à k dimensions par $b = \delta_1 \times \ldots \times \delta_k$ où les δ_i sont des intervalles de valeurs contigües du domaine $dom(d_i)$ de la dimension d_i : $\delta_i \subseteq dom(d_i)$ pour $i = 1, \ldots, k$.*

On notera que, dans le cas où l'on ne spécifie pas un intervalle pour chacune des dimensions du cube, on se ramène à la definition ci-dessus en posant $\delta_i = ALL = dom(d_i)$ pour toute dimension d_i absente de la spécification.

Définition 2 - Recouvrement de blocs. *Deux blocs se recouvrent s'ils ont au moins une cellule en commun.*

Il est facile de voir que deux blocs $b = \delta_1 \times \ldots \times \delta_k$ et $b' = \delta'_1 \times \ldots \times \delta'_k$ du même cube *se recouvrent* si et seulement si pour toute dimension d_i $\delta_i \cap \delta'_i \neq \emptyset$.

On définit maintenant une tranche comme un bloc particulier :

Définition 3 - Tranche d'un cube. *Soit v_i une valeur de la dimension d_i. On appelle* tranche *(ou* slice*) de C associée à v_i, notée $\mathcal{T}_{v_i}$, le bloc $\delta_1 \times \ldots \times \delta_k$ tel que pour tout $j \neq i$, $\delta_j = ALL$ et $\delta_i = \{v_i\}$.*

Une tranche est donc un hyperplan, réduit à une ligne ou une colonne dans le cas particulier d'un cube à deux dimensions. Les notions de support et de confiance associées à un bloc et une valeur de mesure sont définies comme suit :

Définition 4 - Support. *On définit le support d'un bloc de données b dans C pour une valeur de mesure m comme :*

$$supp(b, m) = \frac{\#\ occurrences\ de\ m\ dans\ b}{\#\ cellules\ de\ C}$$

Étant donné un seuil de support σ fixé par l'utilisateur et une valeur de mesure m, un bloc b tel que $supp(b, m) > \sigma$ est appelé σ-fréquent pour m.

On note que le support est anti-monotone, c'est-à-dire que pour tous blocs b, b' et pour tout m :

$$b \subseteq b' \Rightarrow support(b, m) \leq support(b', m)$$

Définition 5 - Confiance. *On définit la confiance d'un bloc de données b pour une valeur de mesure m comme :*

$$conf(b, m) = \frac{\#\ occurrences\ de\ m\ dans\ b}{\#\ cellules\ de\ b}$$

Dans cet article, nous considérons des blocs maximalement spécifiques, c'est-à-dire les blocs définis à partir d'un nombre maximal de dimensions.

Définition 6 - Bloc maximalement spécifique. *Soit σ un seuil de support, m une valeur de mesure, et b un bloc σ-fréquent pour m, b est dit* maximalement spécifique pour m et un seuil de confiance γ *si*

- $conf(b, m) > \gamma$
- *il n'existe pas de bloc b' tel que :*
  - *b' est σ-fréquent pour m*
  - *$\exists j \in [1, k]$ tel que $\delta'_j = ALL$ et $\delta_j \neq ALL$*
  - *$\forall j' \in [1, k]$, $j' \neq j \Rightarrow \delta'_j = \delta_j$*
  - *$conf(b', m) > \gamma$.*

3 Sous-ensembles flous

La théorie des sous-ensembles flous a été introduite par L. Zadeh en 1965 afin de permettre la représentation des connaissances imparfaites [Zadeh, 1965]. Cette théorie offre un cadre formel pour manipuler des données imprécises et/ou incertaines. Par exemple, il est possible de modéliser mathématiquement des données du type *jeune* où un individu appartient *plus ou moins* (de manière graduelle) au concept *jeune*.

De manière générale, un sous-ensemble flou de l'univers X est représenté par sa fonction d'appartenance prenant ses valeurs dans l'intervalle $[0, 1]$. Pour un sous-ensemble flou A de l'univers X, on note μ_A la fonction d'appartenance de A, avec $\mu_A : X \rightarrow [0, 1]$. Pour $x \in X$, $\mu_A(x)$ représente le degré d'appartenance de x au sous-ensemble flou A.

On appelle *support* l'ensemble des valeurs de $x \in X$ telles que $\mu_A(x) > 0$ et *noyau* l'ensemble des valeurs de $x \in X$ telles que $\mu_A(x) = 1$.

On note que tout ensemble classique est un sous-ensemble flou particulier, pour lequel le degré d'appartenance vaut soit 0 soit 1 et non pas toute valeur entre 0 et 1.

4 Génération des blocs

Dans ce travail, nous recherchons les blocs ayant une proportion de cellules de même valeur suffisante. La recherche est fondée sur l'utilisation d'un algorithme par niveaux dérivé des travaux sur APriori [Agrawal *et al.*, 1993], ce type d'algorithmes permettant de proposer des outils efficaces passant à l'échelle. Le but de l'algorithme proposé est de construire les règles les plus spécifiques possibles, c'est-à-dire les règles pour lesquelles la valeur de mesure est déterminée par un maximum de dimensions. Cet algorithme permet également de construire les blocs de taille maximale, en considérant non plus les règles les plus spécifiques mais les règles les plus générales. L'algorithme proposé est le suivant :

Algorithme de recherche des blocs maximalement spécifiques
Entrées : C cube de données défini sur k dimensions, σ seuil de support minimum et γ seuil de confiance minimale.
Sorties : $\mathcal{B}$ l'ensemble des blocs associés au cube C

1. Pour chaque valeur de mesure m du cube C[2]
   (a) Pour chaque dimension d_i (i=1,...,k)
   - $\mathcal{L}_1^i \leftarrow \{v(d_i) \in dom(d_i) | supp(\mathcal{T}_{v(d_i)}, m) > \sigma\}$ où $\mathcal{T}_{v(d_i)}$ est la tranche associée à la valeur $v(d_i)$.
   - Construire les intervalles maximaux $\delta_{i_j} = [\alpha_{i_j}, \beta_{i_j}]$ tels que pour toute valeur $v(d_i)$ située sur d_i entre α_{i_j} et β_{i_j} on a $v(d_i) \in \mathcal{L}_1^i$

   (b) Pour $l = 2$ à k
   - Générer les candidats à partir des fréquents de taille $l-1$. Étant dans le cas de données non binaires, les candidats devront regrouper des intervalles de valeurs sur des dimensions différentes.
     Pour chaque candidat $\delta_{i_1} \times \ldots \times \delta_{i_l}$, considerer le bloc $\delta_1 \times \ldots \times \delta_k$ où $\delta_p = \delta_{p_j}$ si la dimension d_p a été traitée et $\delta_p = ALL$ sinon.
   - Coupure : Supprimer tous les candidats $\delta_{i_1} \times \ldots \times \delta_{i_l}$ tels qu'il existe $p \in \{1, \ldots, l\}$ tel que $\delta_{i_1} \times \ldots \times \delta_{i_{p-1}} \times \delta_{i_{p+1}} \times \ldots \times \delta_{i_l}$ n'est pas fréquent.
   - Évaluer les supports des blocs candidats et supprimer les candidats non fréquents (support $\leq \sigma$)

   (c) Supprimer les blocs b tels que $conf(b, m) \leq \gamma$
2. $\mathcal{B} \leftarrow$ { ens. des blocs engendrés }
3. Si le cube de données doit être visualisé[3], pour chaque bloc, remplacer l'ensemble des valeurs de cellules par la valeur de mesure correspondante. Les cellules ne faisant partie d'aucun bloc sont mises à la valeur nulle. Il est important de noter qu'il peut se trouver plus d'une valeur dans une cellule.

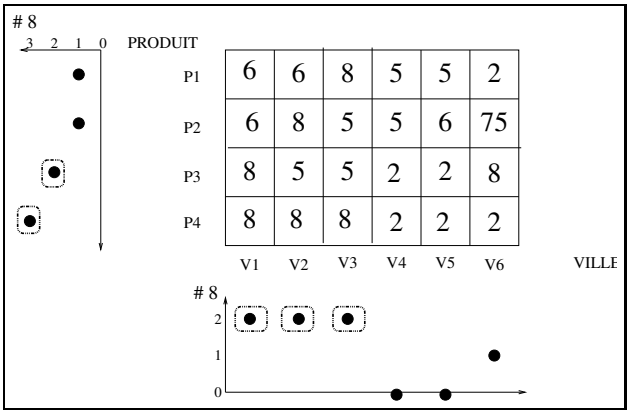

FIG. 2 – Fréquences d'apparition de la valeur **8**

[2]Dans la pratique, par souci d'efficacité, un seul passage sur le cube est effectué pour toutes les valeurs de dimension de manière parallèle.

[3]Dans le cas contraire, il est inutile de procéder au remplacement des valeurs, seules les règles produites étant présentées à l'utilisateur.

La Fig. 2 illustre ce processus pour la valeur de mesure 8 avec un seuil de support minimum fixé à 1/12 (soit 2 cellules dans notre cube en contenant 24) et une confiance fixée à 50% (seuils stricts). La première étape consiste à identifier les valeurs de dimensions correspondant aux tranches où se trouvent plus de deux fois la valeur 8. Sur cet exemple, on trouve $P3$ et $P4$ sur la dimension $PRODUIT$ formant l'intervalle $[P3, P4]$. De même, on trouve $V1$, $V2$ et $V3$ sur la dimension $VILLE$ formant l'intervalle $[V1, V3]$. Ainsi, un seul bloc résulte du croisement de ces régions : $[P3, P4] \times [V1, V3]$, de support suffisant (4/24). Les dimensions ont toutes été exploitées. Il n'y a qu'un bloc maximal restant : $[P3, P4] \times [V1, V3]$ qui a une confiance suffisante puisqu'il contient quatre fois la valeur 8 parmi les 6 valeurs ($66\% > 50\%$). Ce bloc est donc construit et toutes les valeurs qu'il contient sont fixées à 8. La Fig. 3 illustre le résultat de ce processus pour toutes les valeurs de mesure $(2, 5, 6, 8, 75)$. Les quatre blocs suivants sont construits :

- $b_1 = [P1, P2] \times [V1, V1]$ sur la valeur 6
- $b_2 = [P3, P4] \times [V1, V3]$ sur la valeur 8
- $b_3 = [P1, P3] \times [V3, V4]$ sur la valeur 5
- $b_4 = [P3, P4] \times [V4, V6]$ sur la valeur 2

On note que le seuil de support détermine la taille minimale des blocs tandis que le seuil de confiance détermine l'homogénéité à l'intérieur des blocs. En effet, pour une valeur de seuil de support donnée σ, si on note N le nombre de cellules du cube, un bloc ne peut être fréquent que s'il contient au moins $\sigma * N$ cellules. D'autre part, pour une valeur de seuil de confiance donnée γ, un bloc de cardinalité M n'est retenu que s'il contient au moins $\gamma * M$ cellules contenant la valeur de mesure m par rapport à laquelle les calculs sont effectués.

L'algorithme ci-dessus peut facilement être adapté pour construire les *blocs maximaux* (règles les moins spécifiques). Il suffit pour celà de calculer à chaque étape la confiance associée aux blocs et de stopper le parcours des dimensions pour un bloc dès qu'il atteint un niveau de confiance suffisant.

Notre méthode peut être vue comme une méthode de segmentation. Nous sommes conscients que la méthode que nous proposons ne permet pas toujours de retrouver tous les blocs de données. Cependant, cette méthode est efficace pour détecter les blocs de données homogènes les plus pertinents.

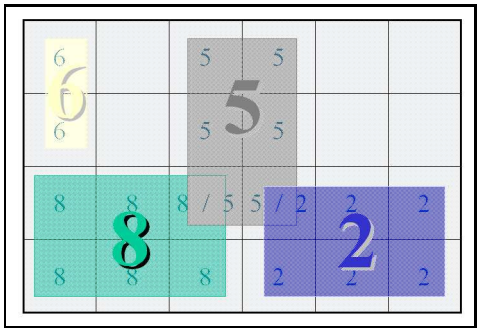

FIG. 3 – tous les blocs

5 Calcul des recouvrements de blocs

Les blocs découverts par notre méthode peuvent se recouvrir mutuellement. Nous explicitons dans cette section la méthode utilisée pour détecter de tels recouvrements.

Cette méthode est fondée sur le constat que deux blocs se recouvrent *si et seulement s*'ils se recouvrent sur *toutes* les dimensions, c'est-à-dire ssi *tous* les intervalles les décrivant ont une intersection non vide. L'algorithme proposé est le suivant :

Algorithme de calcul des recouvrements
Entrées : n blocs $b_j = \delta_{1,j} \times \ldots \times \delta_{k,j}$ (avec $j \in [1, n]$)
Sorties : n groupes de blocs B_j recouvrant b_j (avec $j \in [1, n]$)
- Pour chaque bloc b_j $(j = 1, \ldots, n-1)$
  - $B_j \leftarrow \{b_{j'}\}_{j' \neq j}$
  - Pour chaque bloc $b_{j'}$, $j' \neq j$
    - Pour chaque dimension d_i
      Si $\delta_{i,j} \cap \delta_{i,j'} = \emptyset$ Alors $B_j \leftarrow B_j \setminus \{b_{j'}\}$ et passer au bloc suivant

Dans notre exemple (voir Fig. 3), on obtient les groupes de blocs suivants : $B_1 = \emptyset$, $B_2 = \{b_3\}$, $B_3 = \{b_2, b_4\}$, $B_4 = \{b_3\}$. On note que l'ordre dans lequel les dimensions sont considérées ne modifie pas le résultat mais peut influer sur l'efficacité de la méthode. Par exemple, si deux blocs se recouvrent sur toutes les dimensions sauf une, il est plus efficace de faire la recherche de recouvrement sur cette dernière dimension d'abord (l'algorithme découvrant alors tout de suite qu'il n'y a pas recouvrement) plutôt que de parcourir toutes les dimensions qui se recouvrent en premier lieu.

6 Production des règles et fuzzification

Quand un bloc b_j ne recouvre aucun autre bloc ($B_j = \emptyset$), la règle produite est du type : *Si* $d_1 = \delta_{1,j}$ *et* ...*et* $d_k = \delta_{k,j}$ *Alors* m_j où m_j est la valeur de mesure associée et où les ensembles $\delta_{i,j}$ sont exprimés à l'aide de clauses disjonctives. Par exemple, sur la Fig. 3, la règle produite pour le bloc b_1 correspondant à la valeur 6 est la suivante :

Si la ville est $V1$ *et le produit est* $P1$ *ou* $P2$ *Alors la valeur des cellules est* 6

En cas de recouvrement, notre méthode a recours à des règles floues pour exprimer l'imprécision de la définition des blocs. Les fonctions d'appartenance des sous-ensembles flous sont construites de manière automatique. Plusieurs méthodes de construction des fonctions d'appartenance sont possibles. La méthode proposée ici consiste à associer à chacune des valeurs de l'intervalle l'inverse du nombre de blocs se recouvrant sur cette valeur. Pour les valeurs de dimensions pour lesquelles il n'y a pas de recouvrement, le degré est donc 1 et si trois blocs se recouvrent, le degré est $\frac{1}{3}$.

Soit $b_j = \delta_{1,j} \times \ldots \times \delta_{k,j}$ un bloc. On note, pour $i = 1, \ldots, k$, l'intervalle $\delta_{i,j} = [\alpha_{i,j}, \beta_{i,j}]$.

Algorithme de calcul des sous-ensembles flous

- Pour chaque groupe de blocs $B_j = \{b_{j_1}, \ldots, b_{j_l}\}$ avec $b_{j_p} = \delta_{1,j_p} \times \ldots \times \delta_{k,j_p}$ et $\delta_{r,j_p} = [\alpha_{r,j_p}, \beta_{r,j_p}]$ $(r = 1, \ldots, k)$
  - Pour chaque dimension d_i
    - pour chaque $q = 1, \ldots, l$
      $\alpha_{i,j_q} \leftarrow max(\alpha_{i,j}, \alpha_{i,j_q})$
      $\beta_{i,j_q} \leftarrow min(\beta_{i,j}, \beta_{i,j_q})$
      Ordonner l'ensemble $\{\alpha_{i,j_q} \mid q = 1, \ldots, l\} \cup \{\beta_{i,j_q} \mid q = 1, \ldots, l\}$
      Soit $Int_{i,j} = \{[\alpha^1_{i,j}, \beta^1_{i,j}], \ldots, [\alpha^r_{i,j}, \beta^r_{i,j}]\}$ l'ensemble de tous les sous-intervalles de $[\alpha_{i,j}, \beta_{i,j}]$ ainsi obtenus
    - Associer à $[\alpha_{i,j}, \beta_{i,j}]$ la fonction d'appartenance $\Gamma_{i,j}$ définie par :
      Pour chaque sous-intervalle $[\alpha^s_{i,j}, \beta^s_{i,j}] \in Int_{i,j}$
      Pour chaque valeur v de $dom(d_i)$:

$$\Gamma_{i,j}(v) = \begin{cases} 0 & \text{si } v \notin [\alpha^s_{i,j}, \beta^s_{i,j}] \\ \frac{1}{(1+n_s)} & \text{sinon. } n_s : \text{nbre de blocs } b_{j_p} \text{ de } B_j \text{ t.q. } [\alpha^s_{i,j}, \beta^s_{i,j}] \subseteq [\alpha_{i,j_p}, \beta_{i,j_p}] \end{cases}$$

On considère le bloc $b_3 = [P1, P3] \times [V3, V4]$ avec $B_3 = \{b_2, b_4\}$. Pour la dimension $PRODUIT$, l'intervalle $\delta_{PRODUIT,3} = [P1, P3]$ est découpé en deux sous-intervalles $[P1, P2]$ et $[P3, P3]$. On raisonne de même sur la dimension $VILLE$ et on obtient :

$$\Gamma_{PRODUIT,3}(P) = \begin{cases} 1 & \text{si } P \in [P1, P2] \\ \frac{1}{3} & \text{si } P \in [P3, P3] \\ 0 & \text{sinon} \end{cases} \qquad \Gamma_{VILLE,3}(V) = \begin{cases} \frac{1}{2} & \text{si } V \in [V3, V3] \\ \frac{1}{2} & \text{si } V \in [V4, V4] \\ 0 & \text{sinon} \end{cases}$$

On remarque ici que les deux sous-intervalles $[V3, V3]$ et $[V4, V4]$ pourraient être regroupés en un seul intervalle $[V3, V4]$. Il s'agit là d'une phase d'optimisation facile à mettre en place au cours de l'implémentation que nous ne décrivons pas ici. La Fig. 4 illustre les fonctions d'appartenance construites pour chacune des règles.

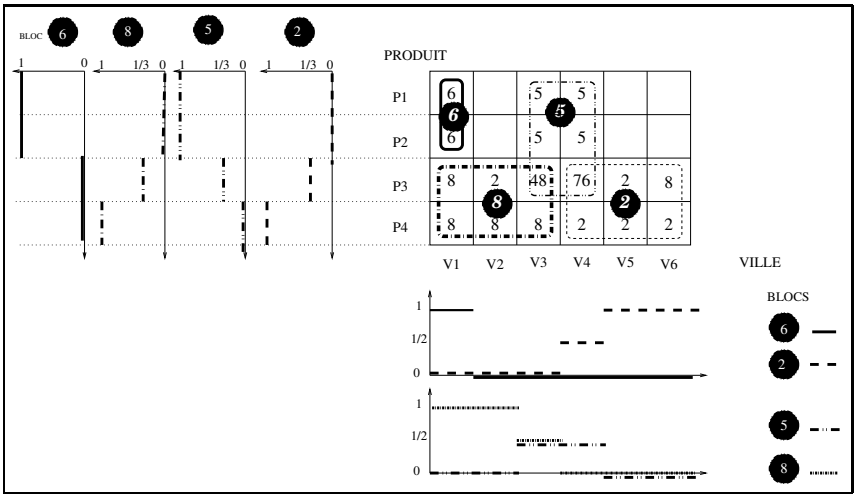

FIG. 4 – Construction des sous-ensembles flous

7 Qualité des représentations

Dans [Choong *et al.*, 2003] une étude des différentes manières de représenter un cube ont ete étudiées. Il est en particulier montré dans cet article que certaines représentations des données sont plus pertinentes que d'autres puisqu'elles permettent de rapprocher des informations et de déduire ainsi des connaissances sur les données. Dans le présent article, nous considérons comme intéressants les rapprochements consistant à regrouper les valeurs de mesure identiques. Il existe d'autres possibilités d'organisations intéressantes, par exemple décrit dans [Choong *et al.*, 2003], où les données sont organisées de telle sorte que la mesure est rangée en ordre croissant le long de toutes les dimensions. Cependant, il est très difficile d'organiser automatiquement les cubes de données d'une manière pertinente. Des méthodes existent, issues des statistiques notamment, mais leur complexité ne permet pas d'envisager leur application sur les données issues des entrepôts de données.

Dans cet article, l'organisation des données n'est pas modifiée avant la construction des règles. Il serait bien sûr intéressant d'organiser le cube afin que les blocs de données soient les plus grands possibles et se recouvrent le moins possible. Mais cette tâche ne constitue pas le but de nos travaux présents. Cependant, il est également intéressant de considérer le problème inverse et d'évaluer la qualité de la représentation à partir des règles construites. Par qualité de la représentation, on entendra *représentation groupée selon les valeurs de cellule*. Cette qualité s'exprime donc en fonction :

- de la proportion de cellules incluses dans des blocs (plus cette proportion est importante, moins il y aura de données non concernées par les règles construites),
- du nombre de blocs construits (plus il y a de blocs, plus les données sont hétérogènes),
- du nombre de blocs par rapport au nombre de valeurs de mesure (retrouver plusieurs blocs correspondant à la même valeur signifie que cette valeur n'est pas bien rangée de manière contigüe),
- du nombre de recouvrements entre blocs et de leur taille (plus les blocs se recouvrent, plus les données sont mélangées).

8 Travaux connexes

Le domaine de la segmentation d'images, et de la segmentation d'images en régions floues, peut paraître proche des travaux présentés ici [Philipp-Foliguet, 2000]. Si les buts sont effectivement assez proches, il n'est pas possible d'utiliser les méthodes développées dans ce contexte de manière directe pour notre problématique. En effet, les techniques de segmentation d'images nécessitent le chargement de toutes les données en mémoire centrale et ne fournissent pas des outils assez performants face aux gros volumes de données auxquels nous devons faire face.

Des méthodes de segmentation (*clustering*) ont été proposées dans le cadre multidimensionnel [Agrawal *et al.*, 1998, Ester *et al.*, 1998]. Dans [Gyenesei et Teuholsa, 2003], les auteurs étudient la production de partitions floues de dimensions numériques. Cependant, dans ces travaux, le partitionnement des données n'intervient pas en fonction de la valeur de mesure, ce qui diffère fondamentalement de notre approche.

Dans [Lakshmanan *et al.*, 2002], les auteurs considèrent la valeur de la mesure (valeur de l'agrégation des données sources), mais se situent dans le contexte du résumé et de la compression de données. Ces travaux ne considèrent pas les aspects liés aux représentations des cubes et ne traitent pas de la recherche de blocs homogènes.

9 Conclusion

Dans cet article, nous proposons une méthode de résumé et de visualisation de données d'une manière efficace. Dans cette approche, des blocs de données identiques sont construits. L'imprécision des règles produites due aux recouvrements de ces blocs est prise en charge par l'utilisation de la théorie des sous-ensembles flous. Si plusieurs autres méthodes existent pour identifier de tels blocs, issues du traitement d'images par exemple, il est cependant crucial de considérer la taille volumineuse des bases des données traitées ici (impossibles à gérer en mémoire centrale) et leur nature multidimensionnelle particulière. Ces autres méthodes ne peuvent donc pas être utilisées de manière directe.

Les perspectives essentielles de ce travail concernent :

- la prise en considération du voisinage des cellules pour la construction des intervalles dans l'algorithme de recherche des blocs,
- l'implémentation de notre méthode,
- l'introduction du flou au niveau de la mesure afin de déterminer des blocs de valeurs semblables et non des blocs de la même valeur, par exemple pour créer un bloc de données correspondant à *environ 5*,
- l'étude d'algorithmes de pré-traitements afin d'organiser les cubes d'une manière pertinente avant d'y chercher les blocs.

Remerciements

Les auteurs remercient l'Ambassade de France de Kuala Lumpur en Malaisie pour son aide et son soutien financier afin de mener à bien cette collaboration et ce travail.

Références

[Agrawal *et al.*, 1993] R. Agrawal, T. Imielinski, et A. Swami. Mining association rules between sets of items in large databases. In *Proceedings of ACM SIGMOD*, pages 207–216, 1993.

[Agrawal *et al.*, 1998] R. Agrawal, J. Gehrke, D. Gunopulos, et P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In *Proceedings of ACM SIGMOD*, pages 94–105. ACM Press, 1998.

[Choong *et al.*, 2003] Y. W. Choong, D. Laurent, et P. Marcel. Computing appropriate representation for multidimensional data. *Data and Knowledge Engineering International Journal*, 45 :181–203, 2003.

[Codd *et al.*, 1993] E.F. Codd, S.B. Codd, et C.T. Salley. Providing olap to user-analysts : An it mandate. Technical report, Arbor Software Corporation, 1993.

[Ester *et al.*, 1998] M. Ester, H.-P. Kriegel, J. Sander, M. Wimmer, et X. Xu. Incremental clustering for mining in a data warehousing environment. In *Proc. 24th Int. Conf. on Very Large Data Bases*, pages 323–333, New York, 1998.

[Ganti *et al.*, 1999] V. Ganti, J. Gehrke, et R. Ramakrishnan. Cactus : Clustering categorical data using summaries. In *Proceedings of ACM SIGKDD, International Conference on Knowledge Discovery and Data Mining*, 1999.

[Gyenesei et Teuholsa, 2003] A. Gyenesei et J. Teuholsa. Multidimensional partitioning of attribute ranges for mining frequent fuzzy patterns. In *Proceedings of FIP'2003*, 2003.

[Inmon, 1992] B. Inmon. *Building the Data Warehouse*. John Wiley & Sons, 1992.

[Lakshmanan *et al.*, 2002] L. Lakshmanan, J. Pei, et J. Han. Quotient cube : How to summarize the semantics of a data cube. In *Proceedings of VLDB'2002*, pages 778–789, 2002.

[Laurent, 2003] A. Laurent. A new approach for the generation of fuzzy summaries based on fuzzy multidimensional databases. *International Journal Intelligent Data Analysis*, 7(2) :155–177, 2003.

[Philipp-Foliguet, 2000] S. Philipp-Foliguet. Segmentation d'images en régions floues. In *Actes des Rencontres francophones sur la logique floue et ses applications*, pages 189–196, 2000.

[Vassiliadis et Sellis, 1999] P. Vassiliadis et T. Sellis. A survey of logical models for olap databases. *SIGMOD Record Journal*, 28(4), 1999.

[Yu *et al.*, 1998] D. Yu, S. Chatterjee, et G. Sheikholeslami. Efficiently detecting arbitrary shaped clusters in very large datasets with high dimensions. Technical report, Department of Computer Science and Engineering, SUNY Buffalo, 1998.

[Yu et Zhang, 2003] D. Yu et A. Zhang. Clustertree : Integration of cluster representation and nearest neighbor search for large datasets with high dimensionality. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 14(3), 2003.

[Zadeh, 1965] L. Zadeh. Fuzzy sets. *Information and Control*, (8) :338–353, 1965.

Summary

In the context of multidimensional data, OLAP tools are appropriate for the navigation in the data, aiming at discovering pertinent and abstract knowledge. However, due to the size of the dataset, a systematic and exhaustive exploration is not feasible. Therefore, the problem is to design automatic tools to ease the navigation in the data and their visualization. In this paper, we present a novel approach allowing to build automatically blocks of similar values in a given data cube and to associate these blocks with rules. Our method is based on a level-wise algorithm (a la Apriori) and on the theory of fuzzy sets. The latter is considered here due to the fact some of the blocks computed by our algorithm can overlap.

Keywords : Multidimensional databases, Level-wise algorithms, Fuzzy rules, Data visualization.