**Master 2 MIAGE CFA**

2010-2011

**Base de données multi-dimensionnelles et OLAP**

**Projet n 2 : Règles d’associations**

**Florian GOUIN et Arnaud BRETON**

Sommaire

[Introduction 3](#_Toc283594430)

[Présentation du projet 4](#_Toc283594431)

[Conception générale 4](#_Toc283594432)

[Choix technologiques 4](#_Toc283594433)

[Répartition du travail 5](#_Toc283594434)

[Interface graphique 5](#_Toc283594435)

[Analyse de performance 6](#_Toc283594436)

[Conclusion 8](#_Toc283594437)

# Introduction

L’objet de projet de dernière année de master 2 MIAGE est d’appréhender les règles d’associations de manière pratique via une implémentation informatique.

Le but de cette implémentation est de permettre à un utilisateur de connaître l’ensemble des règles d’associations intéressantes, c'est-à-dire celle dépassant une certaine confiance, afin d’établir des liens entre des éléments à priori sans aucune connexion.

Cette méthode est utilisée pour analyser les comportements des consommateurs et ainsi dégager des tendances (par exemple, que les consommateurs achetant du lait, achètent aussi de la bière).

Dans notre implémentation, nous partons d’un tableau disjonctif complet, à savoir une table binaire (ne contenant donc que des 0 et des 1) pour calculer, dans un premier temps les ensembles dits fréquents (ceux qui dépassent un seuil donné) puis, à partir de ces ensembles, nous calculons les règles d’associations intéressantes (celles qui dépassent un seuil donné).

# Présentation du projet

Comme nous l’avons vu ci-dessus, l’application permet à l’utilisateur d’obtenir les règles d’associations à partir d’une table binaire (un tableau disjonctif complet).

Pour cela, l’utilisateur doit saisir un seuil de confiance et de support, le nom de la table à exploiter (parmi celles présentes dans la base de données).

Le lancement des calculs recherche alors via l’algorithme A-Priori l’ensemble des attributs fréquents de la table.

Ce premier retour intermédiaire est affiché par l’IHM.

A partir de cet ensemble, le programme génère l’ensemble des règles possibles () puis calcule leur confiance respective.

Le retour est donné à l’IHM qui affiche l’ensemble des règles intéressantes (celles dont la confiance dépasse le minimum).

Le programme permet aussi à l’utilisateur de générer des tables aléatoires à n colonnes et m lignes (n et m étant définis par l’utilisateur) peuplés de données binaires aléatoires. Ces tables permettent à l’utilisateur de tester les performances de l’algorithme en faisant varier la taille de la table.

## Conception générale

L’application est divisée en quatre couches :

1. *L’Interface graphique*, qui permet à l’utilisateur de saisir les seuils de support et de confiance et de calculer les règles d’associations d’une table donnée. C’est aussi elle qui réalise les contrôles de surface,
2. *Le modèle*, qui calcule les ensembles fréquents et règles d’associations,
3. *La couche d’accès aux données* (CAD) qui interroge la base de données,
4. *La base de données* qui contient les tables sur lesquelles on veut extraire les règles d’associations.

Nous avons donc mis en place, côté applicatif (IHM, modèle, CAD), un découpage du type Modèle-Vue-Contrôleur (MVC).

La communication entre la Vue (l’interface graphique) et le modèle est assurée par le patron de conception *« Observateur »* (<http://fr.wikipedia.org/wiki/Observateur_(patron_de_conception)>)

## Choix technologiques

Le projet a entièrement été réalisé en Java (implémentation de l’algorithme A-Priori, l’interface graphique et JDBC (*Java DataBase Connectivity,* qui est une interface de programmation standard pour l’accès aux bases de données) pour l’accès à la base de données *MySQL*).

L’interface graphique a été réalisée grâce aux librairies graphiques SWT (*Standard Widget Toolkit,* <http://www.eclipse.org/swt/>)

La gestion des sources est assurée par SVN (SubVersioN), hébergé par Google Codes.

## Répartition du travail

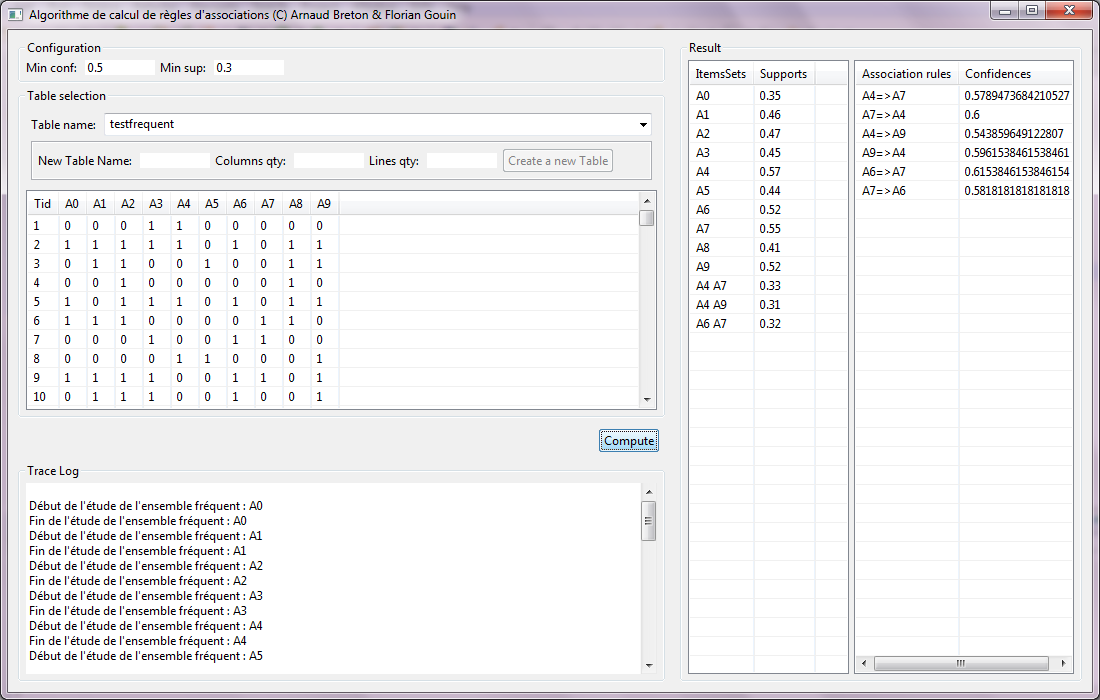
L’ensemble de l’interface graphique a été réalisée par Florian GOUIN.

Le moteur de calcul des règles d’associations a été réalisé par Arnaud BRETON.

Bien sûr, chacun d’entre nous est intervenu de manière ponctuelle sur la partie de l’autre afin de donner son point de vue ou d’apporter son aide sur un sujet délicat (point de conception (communication entre le modèle et l’IHM par exemple), écriture d’algorithmes performants, etc.).

## Interface graphique

Voici une vue de l’interface graphique de notre application.



Elle se décompose en 4 parties :

* Dans la première partie identifiée par l’encadré *« Configuration »* nous retrouvons 2 champs Min Conf et Min Sup. Le premier correspond au minimum de confiance tandis que le second correspond au minimum de support définis tous deux pour l’algorithme de calcul de règles d’association. Un contrôle de saisie permet à l’application de vérifier que les champs saisis sont correct c'est-à-dire ne comportant que des chiffres et des virgules et étant compris entre 0 et 1. Si la valeur était comprise hors de ce domaine la saisie serait alors transformée en sa borne la plus proche. (ex : 99 -> 1)
* Dans le deuxième encadré intitulé *« Table selection »*, nous retrouvons une liste déroulante permettant de choisir une table dans la base de données. Si aucune table n’est disponible ou que l’on souhaite créer une nouvelle table, l’encadré situé juste en dessous va nous permettre d’arriver à nos fins. Trois éléments sont alors paramétrables, il s’agit du nom de la nouvelle table créée, ainsi que du nombre de colonnes pour celle-ci mais aussi du nombre de ligne. Ici encore des contrôles de saisie permettent de s’assurer de la pertinence des données saisies. De plus si les données sont incorrect, le bouton « Create a new table » devient non cliquable. Dans le cas contraire un ‘clic’ sur ce bouton lancera une requête de génération de table et un algorithme de peuplement remplira la table de 0 et de 1 de façon aléatoire. Il s’agira donc d’une table binaire.
* Le troisième encadré « Trace log » permet de surveiller les calculs de l’application. Tout algorithme lancé fournira des informations en direct sur son activité dans cet encadré. Comme nous pouvons l’observer ici, les débuts et fins d’étude de l’ensemble des fréquents sont affichés en direct. Cela permet de savoir à n’importe quel moment ce que fait l’application.
* Le quatrième et dernier encadré *« Result »* se décompose en 2 tableaux. Ceux-ci affichent le résultat de l’algorithme de calcul des règles d’associations suite à un ‘clic’ sur le bouton *« Compute »*. Le tableau de gauche nous fournit pour chaque groupe retenu la valeur de son support. Comme nous pouvons l’observer dans cet exemple, les seuls groupes retenus sont ceux ayant une valeur de support supérieure à 0.3. Le tableau de droite ensuite nous indique l’ensemble des règles d’association retenues ainsi que leur confiance. Ici encore notre exemple nous montre bien que les seules règles d’associations retenues sont celles ayant une valeur de confiance supérieure à 0.5.

## Analyse de performance

La démarche suivie pour réaliser cette analyse a été la suivante :

* Génération d’une table à n lignes, m colonnes (n compris entre 10 et 460 et m compris entre 5 et 50)
* Application de l’algorithme sur ces tables avec un minimum de support et de confiance fixe pour une itération donnée (allant de 0.1 à 1)
* Observation des résultats

Les résultats de l’analyse sont les suivants :

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Taille (LxC)/ Seuils (support, confiance)**  **Temps en ms** | **0.3/0.5** | **0.4 / 0.6** | **0.5/0.7** | **0.6 / 0.8** | **0.7 / 0.9** | **0.8/1.0** | **0.9/1.0** | **1.0/1.0** |
| 5\*10 | 90 | 75 | 20 | 36 | 21 | 19 | 17 | 21 |
| 10\*60 | 112 | 43 | 19 | 22 | 20 | 17 | 19 | 19 |
| 15\*110 | 186 | 48 | 42 | 18 | 28 | 28 | 20 | 19 |
| 20\*160 | 213 | 100 | 55 | 31 | 22 | 32 | 23 | 30 |
| 25\*210 | 323 | 138 | 62 | 23 | 30 | 28 | 25 | 21 |
| 30\*260 | 372 | 302 | 59 | 23 | 24 | 24 | 26 | 31 |
| 35\*310 | 345 | 203 | 81 | 43 | 35 | 30 | 28 | 39 |
| 40\*360 | 493 | 255 | 102 | 50 | 37 | 34 | 24 | 98 |
| 45\*410 | 726 | 369 | 111 | 41 | 38 | 35 | 33 | 65 |
| 50\*460 | 660 | 483 | 120 | 32 | 45 | 43 | 32 | 27 |

On observe que plus le support et la confiance sont élevés, plus les temps d’exécution sont faibles. Ceci s’explique par le fait que l’ensemble des fréquents est de plus en plus réduit (il y a de moins en moins d’attributs dont le support dépasse le seuil). Les règles qu’il est alors possible de générer à partir de cet ensemble est donc lui aussi réduit ainsi que celles dont la confiance dépasse le seuil.

Globalement, nous observons tout de même de très bonnes performances de l’algorithme, dont le temps maximal d’exécution ne dépasse par la seconde (726ms).

Nous avons exécuté l’algorithme sur une table plus volumineuse (50 000 lignes, le nombre de lignes variant toujours dans l’espace décrit ci-dessus) et nous obtenons là aussi, au regard du volume de donnée, des performances acceptables, autour de 20 secondes pour le maximum et la seconde pour le minimum.

# Conclusion

Ce projet nous a permis de mettre en pratique un algorithme vu en cours, en l’implémentant complètement sur machine.

Il nous a permis de comprendre l’intérêt économique qui est induit par les règles d’associations qui vont permettre aux analystes de déduire des règles qui ne paraissent par évidente lors de l’observation des tables d’une base, surtout quand celle-ci comporte un grand nombre d’entrées.

Il nous a aussi sensibilisé aux problématiques d’optimisation. En effet, la complexité de ce type d’algorithme fait qu’il est assez compliqué d’être optimal. L’algorithme implémenté dans le cadre du projet n’est pas le plus optimal : d’autres algorithmes (y compris une variante de l’ « A-Priori » basé sur la théorie des arbres) plus adaptés à des bases de données plus volumineuses existent.