Visualisation des règles d'association en environnement virtuel 3D interactif

Dominique Hervouet

Conservatoire National des Arts et Métiers
Centre régional associé des pays de la Loire

Mémoire
présenté en vue d'obtenir
le diplôme d'ingénieur en INFORMATIQUE
Option systèmes d'information
par

Dominique HERVOUET

Soutenu le 18 février 2011

Visualisation des règles d'association en environnement virtuel 3D interactif

Jury

Présidente : Mme MÉTAIS, *professeur CNAM PARIS*
M. BRIAND, *professeur, Ecole Polytechnique de l'université de Nantes*
M. GUILLET, *maître de conférences, Ecole polytechnique de l'université de Nantes*
Melle BEN SAID, *doctorant, Ecole polytechnique de l'université de Nantes*
M. LASTENNET, *responsable département informatique CNAM NANTES*

À mon père.

Remerciements

Je remercie toutes les personnes qui m'ont permis l'accomplissement de ce mémoire : M. le professeur Henri Briand et M. Fabrice Guillet qui m'ont accueilli au sein de leurs équipes de recherche, ainsi que Melle Zohra Ben Saïd pour ses corrections appliquées. Je remercie aussi et particulièrement Mr Jean-Louis Brunel, responsable de la filière informatique du centre associé d'Aix en Provence, pour ses encouragements, certes, parfois directifs, mais non moins stimulants sur l'ensemble du cursus de formation CNAM.

Table des matières

Remerciements 3
Introduction 7
1.1 Présentation 7
1.2 Problématique 8
1.3 Objectifs de réalisation 9
1.4 Contexte du stage 9
1.5 Organisation 10
Etat de l'art 11
2.1 L'Extraction de Connaissances dans les Données 11
2.1.1 Enjeux 11
2.1.2 Déroulement général d'une fouille de données 13
2.1.3 Vue d'ensemble des techniques d'extraction 14
2.2 Les règles d'association 15
2.2.1 Modélisation d'une règle d'association 15
2.2.2 Propriété de fréquence des ensembles d'*items* 16
2.3 Mesures d'intérêt 17
2.3.1 Indices de support et de confiance 17
2.3.2 Mesures d'intérêt objectives 19
2.3.3 Mesures d'intérêt subjectives 24
2.4 Algorithmes d'extraction 25
2.4.1 Algorithmes exhaustifs 25
2.4.2 Algorithmes à contraintes 29
2.5 Post-traitement en ECD 30
2.5.1 Exploration textuelle 30
2.5.2 Langage de requêtes 31

2.5.3 Explorations graphiques 31
2.6 Visualisation de l'information 35
2.6.1 Modélisation des outils 35
2.6.2 Sémiologie 38
2.6.3 Représentations graphiques 2D *vs* 3D 38
2.7 Réalité Virtuelle 39
2.7.1 Définition, concept et enjeux 39
2.7.2 Apport de la réalité virtuelle à l'ECD 41
2.7.3 Visualisation 3D de l'information 41
2.7.4 Exemple de visualisation de règles d'association : *ARVis* 44
Etude préalable 47
3.1 Cahier des charges 47
3.1.1 Besoins 47
3.1.2 Modèle de données 48
3.1.3 Environnement 49
3.1.4 Contraintes 49
3.2 Etude des solutions techniques 50
3.2.1 Choix du Système de Gestion de Base de Données (SGBD) 50
3.2.2 Extraction des règles 54
3.2.3 Technologie 3D 54
3.2.4 Langage de programmation et environnement de développement 59
3.2.5 Récapitulatif des choix techniques 59
3.3 Proposition d'une nouvelle métaphore 3D 60
3.3.1 Présentation de la métaphore 60
3.3.2 Calcul des indices de qualité 61
3.3.3 Exemple explicatif 62
3.3.4 Encodage graphique des mesures d'intérêt 65
Réalisation 73
4.1 Présentation 73
4.2 Organisation 73
4.2.1 Dates clés 73
4.2.2 Planification des tâches 74
4.3 Contexte technique 74
4.4 Normes et outils 75
4.5 Spécification 76
4.5.1 Diagramme de classe simplifié 76

4.5.2 Description des classes principales : 76
4.5.3 Interface utilisateur 84
4.5.4 Diagramme d'activité : interface 85
4.5.5 Diagramme d'activité : extraction des règles (*class* Main_Prog) 86
4.5.6 Comparaison des résultats d'extraction avec Tanagra 87
4.6 Placement du graphe de prémisse 88
4.6.1 Initialisation de l'algorithme de placement: 88
4.6.2 Echelle graphique 88
4.6.3 Corrélations graphiques ente le lift et les distances inter-sphères 89
4.7 Expérimentation sur des données réelles 94
4.7.1 Efficacité du système d'extraction 94
Conclusion 96
5.1 Acquis 96
5.2 Perspectives 97
Liste des tableaux 99
Liste des figures 99
Bibliographie 101
Lexique 104

Chapitre 1

Introduction

1.1 Présentation

L'homme du XXIe siècle baigne dans un flot d'informations statistiques, résultats économiques, prévisions sur le climat, la population, les ressources, etc. dont il ne voit que l'écume sans en voir les lames de fond. Depuis plus d'un siècle, un large éventail de méthodes d'analyses statistiques ont été proposées afin de soumettre ces jeux de données à des tests non destructifs, à partir d'hypothèses et sur des paramètres particuliers ou pour estimer la validité des modèles de probabilité. La véracité de l'information et les connaissances extraites par ces méthodes, dépendent alors étroitement du niveau de qualité et de l'intégrité des données. En effet, une valeur aberrante ou peu fiable, incohérente ou obsolète peut se propager de façon endémique à tous les types de données stockées et ainsi, contaminer l'interprétation de l'information effectuée par l'analyste. Par conséquent, une prise de décision qui s'appuierait sur des valeurs considérées comme anormales entraine parfois des coûts financiers qui peuvent être considérables[1].

L'Extraction de la Connaissance dans les Données (ECD) apparaît alors comme une alternative pour faire face à ces problèmes déjà anciens. FRAWLEY et *al*., (1992) présente l'ECD comme étant « l'extraction non triviale de connaissances implicites, inconnues au préalable et potentiellement intéressantes contenues dans les données » Il s'agit de rechercher des régularités et des caractéristiques remarquables pour découvrir dans la « gangue » de données, ce que l'on nomme « pépites » de connaissances. Contrairement aux méthodes statistiques dont l'utilisation et l'interprétation des résultats exigeaient des compétences particulières, les moyens de fouille de données en ECD sont élaborés pour être accessibles par des utilisateurs non spécialistes. L'utilisation de l'outil informatique élargit le cercle des utilisateurs par rapport aux techniques statistiques.

[1] Selon le TDWI (*The Data Warehousing Institute*), les coûts engendrés par la non-qualité des données s'élevaient en 2002 à 611 milliard de dollars par an pour l'économie américaine [BER06].

Les méthodes et outils développés pour l'ECD cherchent à combiner les capacités de calcul des ordinateurs avec les potentialités humaines d'analyse et de jugement. Les objectifs de KUNTZ et *al*, (2006), sont d'intégrer l'utilisateur comme une heuristique dans le système de fouille de données.

Au sein des méthodes de l'ECD, les règles d'association permettent de matérialiser les relations de type « prémisse » implique « conclusion ». Les parties droite et gauche sont des groupements d'attributs contenus dans la base de données. Nous présenterons et définirons ce concept à l'origine de l'ECD ainsi que les méthodes et techniques associées pour découvrir de la connaissance utile dans les données.

Cette étude porte principalement sur une problématique en fouille visuelle de données et notamment sur la visualisation de règles d'association. A l'aide de récentes technologies dédiées aux espaces tridimensionnels (3D)[2]. Nous chercherons à offrir à l'utilisateur un outil avec lequel ce dernier devient l'acteur principal du processus de décision.

1.2 Problématique

BLANCHARD (2005) dit que « le nombre de règles obtenu en sortie des algorithmes d'extraction croit de façon exponentielle avec le nombre d'attributs décrivant les données ». Une des problématiques en ECD est alors de faire face à ces gros volumes. Nous verrons que de nombreuses techniques et méthodes ont été développées pour y répondre. En terme de visualisation, les objets graphiques présentés à l'utilisateur sont chargés de traduire les données en information. En ECD, donner du sens graphique aux représentations reste une tâche complexe et un problème ouvert. Nous étudierons cet axe de recherche et les techniques mises en œuvre pour répondre à ce challenge. Par ailleurs, deux tendances récentes pour la fouille de données visent à utiliser :

- une métaphore 3D de règles d'association,

- les techniques d'interaction empruntées au domaine de la réalité virtuelle (RV).

[2] Cette expression caractérise l'espace qui nous entoure, tel que perçu par notre vison en termes de largeur, hauteur, et profondeur.

1.3 Objectifs de réalisation

L'objectif est de fournir un outil de fouille visuelle de données et d'élaborer une nouvelle métaphore de règles d'association. A partir d'un Système de Gestion de Base de Données (SGBD), le programme doit prévoir des fonctionnalités d'extraction locales depuis un client de visualisation. Les règles sont alors induites par les interrogations et les suggestions de l'utilisateur. Il se voit intégré dans la boucle de recherche de la connaissance.

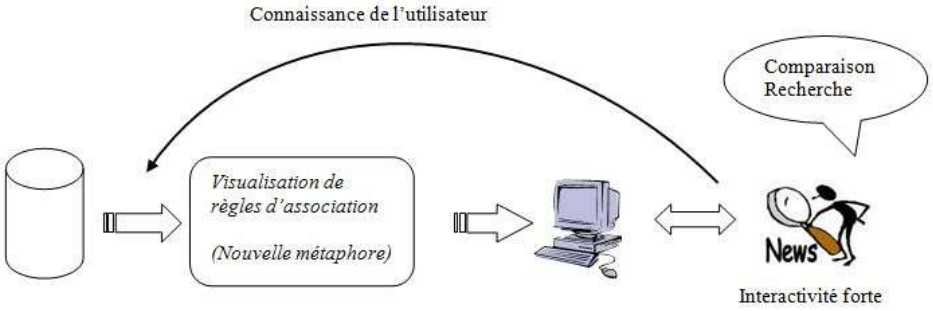


FIG.1 - Pilotage d'une fouille interactive de données

Les règles sont décrites par l'élaboration de mesures qui évaluent suivant le contexte, une notion d'intérêt par rapport aux attentes de l'utilisateur. Nous détaillerons les mesures encodées graphiquement dans notre métaphore 3D. Le choix du SGBD est libre mais doit supporter de gros volumes de données et les traitements relatifs à l'extraction des règles. Le client de visualisation est réalisé par un programme développé en C/C++ et s'appuie sur les couches graphiques d'OpenGL[3].

1.4 Contexte du stage

J'ai effectué ce stage au sein de l'équipe Connaissances et Décisions (COD) de l'UMR (Unité Mixte de Recherche) 6241 du CNRS dans les locaux de l'école Polytech Nantes. Le laboratoire est spécialisé dans les sciences et technologies du logiciel». Il est composé de dix équipes de recherche réparties sur deux thèmes principaux qui sont les architectures distribuées et les systèmes d'aide à la décision. Les encadrants sont Fabrice Guillet (maître de conférences, Ecole Polytechnique de l'université de Nantes), Julien Blanchard (maître de conférences, Ecole polytechnique de l'université de Nantes) et Zohra Ben Saïd (doctorant).

[3] OpenGL (*Open Graphics Library*) spécification d'API (*Application Programming Interface*) multi-plateforme pour la conception d'applications générant des images 3D ou 2D.

1.5 Organisation

Le chapitre 2 présente un état de l'art des domaines qui concernent le projet. Il présente d'une façon générale le processus d'extraction de connaissance. Nous évoquerons ensuite les différentes solutions existantes en visualisation de l'information et leurs applications à la représentation de l'information en ECD.

Le chapitre 3 précise le cadre d'étude. On y analyse les différentes solutions et choix techniques qui pourront répondre aux contraintes de l'outil à développer. Nous détaillerons également la nouvelle métaphore 3D.

Le chapitre 4 détaille la conception technique et la réalisation de l'outil d'extraction et de visualisation des règles d'association. Les performances de l'outil seront évaluées avec des données réelles fournies par Nantes habitat.

Le chapitre 5 ouvre des perspectives tant théoriques que techniques et évoque les nombreux chemins qui restent à explorer...

Chapitre 2

Etat de l'art

2.1 L'Extraction de Connaissances dans les Données

2.1.1 Enjeux

L'émergence des techniques d'Extraction de la Connaissance dans les Données (ECD) est le résultat de l'accroissement de la taille des bases de données. Les données traitées par jour dépassent le milliard[4] avec une puissance de calcul toujours plus importante (loi de Moore[5]). Les entreprises dans un contexte de concurrence accrue qui stockent sur supports informatiques des données informationnelles sur leur client se sont donc rapidement intéressées aux outils utilisés en ECD.

Dans un but économique, les entreprises cherchent à valoriser l'information potentielle et la connaissance qu'elles ont encore non exploitées, car masquées par de trop gros volumes de données. En règle générale, ces techniques visent à découvrir des corrélations existantes dans les jeux de données et à définir des motifs séquentiels dans lesquels, il est possible de tirer de l'information pertinente.

L'une des premières applications des règles d'association fut l'étude du panier de la ménagère dans le domaine de la grande distribution. La recherche de règles d'association est une méthode pour extraire de la connaissance dans les données. Elle consiste à mettre en évidence des combinaisons de produits achetés ensemble dans un supermarché. D'un point de vue marketing, ces règles détectent les comportements et les besoins nouveaux du consommateur. Les jeux de données constitués par les enregistrements des tickets de caisses dissimulent à cet effet des informations utiles sur ses attitudes et les nouvelles tendances, qu'elles soient générales, ou particulières.

[4] http://cpa.enset-media.ac.ma/datamining.htm.
[5] La loi de Moore annonçait en 1965 que la puissance des processeurs doublerait tous les ans pour un même coût. Elle s'est révélé jusqu'ici étonnamment exact et devrait en principe le rester jusqu'en 2015 avant de se confronter réellement aux effets quantiques (bruits parasites).

La connaissance découverte au travers des règles permet aux experts de piloter les actions commerciales ou d'approfondir plus encore l'analyse comportementale du consommateur. L'exemple le plus connu et que l'on trouve le plus souvent dans la littérature énonce qu'il y aurait eu la mise en évidence par les magasins *Wal-Mart*[6], dont l'enseigne est née dans les années 1960 aux Etats-Unis, d'une corrélation très forte entre l'achat de couches pour bébés et de bières le samedi après-midi. Suite à cette information difficile à deviner intuitivement mais bien réelle, l'aménagement des rayons est redéfini en concluant que les couches pour bébés, achat lourd et encombrant sont achetées principalement par les hommes qui en profitent pour s'approvisionner en bière. Dans les années 1990, les revenus de cette entreprise ont quadruplé. Pour la première fois, ils ont atteint la somme de 1 milliard de dollars de chiffre d'affaires en une semaine.

Depuis l'étude du panier de la ménagère, l'ECD trouve chaque jour des applications plus nombreuses pour les secteurs économiques et financiers ou scientifiques et industriels, secteur social, etc. Les secteurs d'activité qui analysent les plus gros volumes de données sont les plus concernés. Trouver dans ces volumes des informations inconnues jusqu'alors dans le but de prédire des comportements ou des événements particuliers n'est pas totalement nouveau, et reste l'objectif de base en ECD.

La fouille de données est une phase importante qui fait partie du processus ECD. Les techniques utilisées dans cette phase (réseaux de neurones[7], réseaux bayésiens[8], les arbres de décisions, les règles d'association, etc.), distinguent l'ECD de l'analyse de données issue des champs de recherche mathématiques et statistiques. L'ECD se trouve au carrefour de nombreuses disciplines comme l'apprentissage automatiques, la reconnaissance de formes, les bases de données, les statistiques, la représentation de la connaissance, l'intelligence artificielle, les systèmes experts, etc.

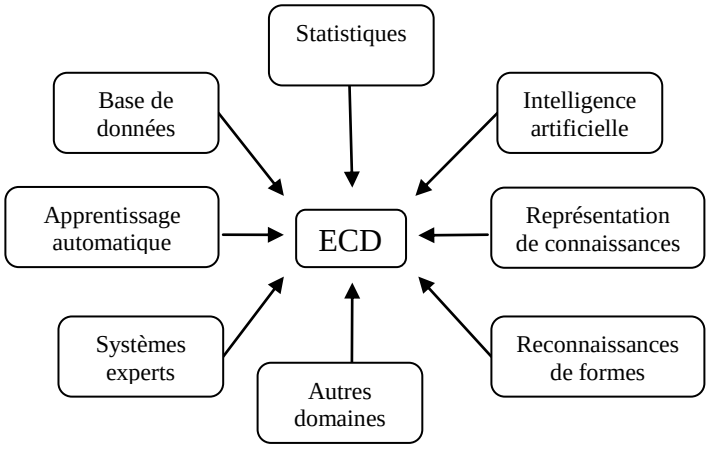


FIG.2 - L'ECD à la confluence de nombreux domaines

[6] http ://www.zdnet.fr/blogs/2005/11/27/datamining/
[7] Les objets sont affectés séquentiellement à des groupes en fonction de leur proximité et le processus d'apprentissage est incrémental.
[8] Les réseaux bayésiens sont des modèles probabilistes graphiques permettant d'acquérir, de capitaliser et d'exploiter des connaissances.

2.1.2 Déroulement général d'une fouille de données

Le processus ECD est généralement découpé en trois phases principales identifiant les actions à effectuer :

- le prétraitement,
- la fouille de données,
- le post traitement.

Nous présentons le détail de ces phases (FIG.3).

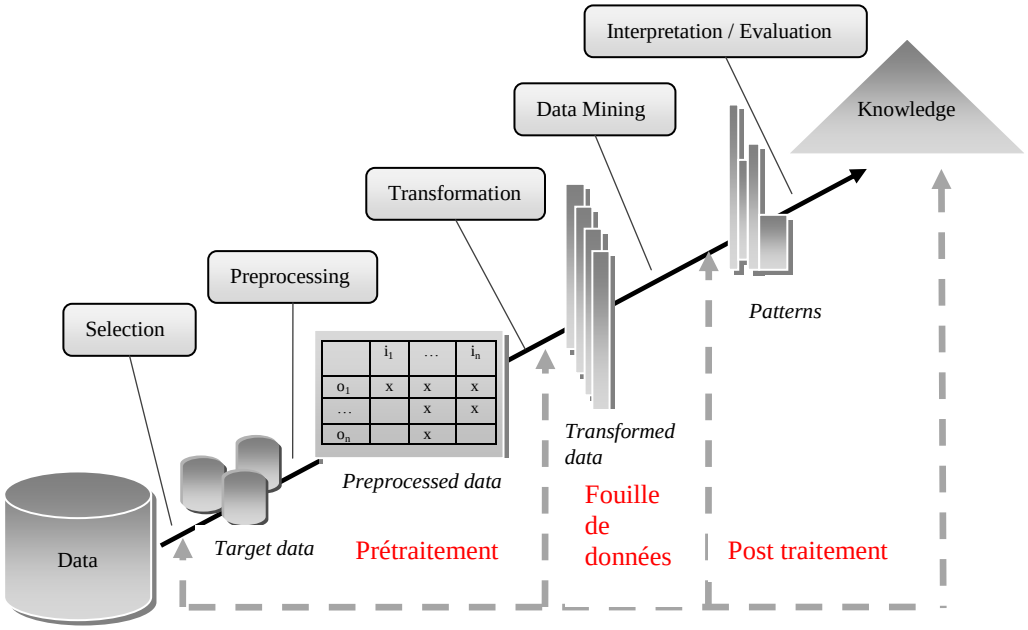


FIG.3 - Processus ECD [Fayyad et *al.*, 1996]

1. Le prétraitement :

   - **Sélection** : Le jeu de données et les attributs pertinents sont extraits des données brutes,
   - **Prétraitement** : Concerne la mise en forme des données souvent de nature différentes (texte, image, vidéo, etc.). Cette opération porte sur l'accès aux données en vue de construire des corpus de données spécifiques. Les valeurs aberrantes ou manquantes sont traitées par l'élimination complète de la ligne ou par interpolation.
   - **Transformation** : De nouveaux attributs sont formés à partir des attributs d'origine, la structure des données est modifiée pour en faciliter la fouille (dénormalisation),

2. La fouille de données :

    - C'est l'étape « moteur » de l'ECD ou les algorithmes sont appliqués. ZIGHED et RAKOTOMALA (2002) identifient deux catégories de méthodes en fouille de données :

        a. Les méthodes de classification et de structuration : L'objectif principal étant de faire face à la profusion de données en identifiant des groupes d'objets semblables, au sens d'une métrique donnée (ex : monothétiques[9], polythétiques[10], ou basées sur des réseaux de neurones),
        b. Les méthodes d'explication et de prédiction : Elles ont pour objectif d'établir à partir des données transformées, un modèle explicatif ou prédictif entre d'une part, un attribut particulier à prédire et d'autre part, des attributs prédictifs. Dans le cas ou un tel modèle est validé, il peut être utilisé à des fins de prédiction (ex : les arbres de décisions, les réseaux bayésiens, les règles d'association, etc.).

3. Le post traitement

    - Cette étape consiste à interpréter et à évaluer les informations par l'expert métier sous forme de listes textuelle ou encodées dans une représentation graphique. Les outils de visualisation offrent alors à l'analyste une vision synthétique de l'ensemble des données (ex : histogramme, nuage de points, graphe de contingence, etc.). Ces méthodes sont issues de la statistique descriptive, de l'analyse de données et des techniques de visualisation graphiques dont certaines font appel à la réalité virtuelle et à des métaphores calquées sur le modèle mental humain.

Notre étude porte sur la phase de post-traitement et traite plus particulièrement une problématique en fouille de données.

2.1.3 Vue d'ensemble des techniques d'extraction

Les techniques d'Extraction de Connaissances dans les Données sont séparées en deux classes, ALESSANDRI M. (2010) :

1) **Les techniques supervisées** (classification) : Elles consistent à choisir au préalable un ou plusieurs attributs ou *items* connus, appelé(s) attribut(s) endogène(s). Puis, à partir de ce choix, à déterminer au sein des autres attributs, les conditions associées aux valeurs particulières prisent par les attributs endogènes. On peut par exemple chercher des éléments de réponse à la question : Quelles sont les conditions (signaux d'état des périphériques d'une architecture numérique complexe) qui ont entrainé l'apparition d'un défaut particulier ? Ces techniques comprennent les arbres de décisions, les

[9] Recherche de partitions sur l'ensemble des objets, dans la classe des vertébrés par exemple, toutes les espèces ont en commun la présence de vertèbres.

[10] Recherche de partitions dans lesquelles les éléments d'une même classe ont, entre eux, une certaine ressemblance, eu égard aux éléments d'autres classes de cette même partition devant être les plus dissemblable possible, au sens d'un critère préétabli.

réseaux bayésiens, les réseaux de neurones et certaines techniques statistiques.

2) **Les techniques non supervisées** (*clustering*) : Elles cherchent à détecter les régularités dans les données et à les classer sans fixer, à priori, d'éléments à découvrir. De nature non supervisées, ces techniques ne nécessitent donc pas de préciser des attributs endogènes et envisagent donc toutes les combinaisons possibles à partir du jeu d'attributs. L'utilisation de ces techniques peut se baser sur les statistiques, les réseaux de neurones, les règles d'association.

Nous observons dans cette répartition, que les réseaux de neurones appartiennent aux deux classes. En effet et suivant les cas d'utilisation, la nature de ces techniques d'extraction peut être supervisée ou non. Il est nécessaire par exemple pour la recherche de règles d'association classée parmi les techniques non supervisées, d'utiliser des variables cibles, correspondant à des seuils appliqués aux mesures d'intérêt qui décrivent les règles. Les résultats obtenus sont alors conditionnés par le choix de ces variables. Une autre méthode existante pour l'extraction de règles génère par essence une arborescence dans le processus de recherche (algorithme Charm [Zaki et Hsiao, 2002]). Cette méthode peut donc s'apparenter aux arbres de décision qui eux sont classés dans les techniques supervisées.

2.2 Les règles d'association

Le processus de fouille de données par la recherche de règles d'association [Agrawal et al, 1993] se base sur une classe particulière de motifs appelés *itemsets* fréquents ou conjonction d'*items* fréquents. En s'appuyant sur la particularité de fréquence des *itemsets*, la technique consiste à mettre en évidence des règles de la forme prémisse (antécédent) → conclusion (conséquence). Les règles d'association expriment alors à partir des données contenues dans une base de données relationnelle les tendances implicatives entre les attributs de la prémisse, et ceux qui apparaissent dans la conclusion.
BLANCHARD (2005), définit cette implication non pas au sens mathématique, mais comme étant la tendance de la conclusion à être « vraie » lorsque la prémisse est « vraie ». La relation d'implication mise en évidence par les connexions existantes entre les données fait alors émerger la relation causale qui amènera la déduction jusqu'alors noyée dans l'ensemble des données.

2.2.1 Modélisation d'une règle d'association

Soit $I = \{i_1, i_2, \ldots i_n\}$ un ensemble d'attributs distincts de la base. $T = \{t_1, t_2, t_3\}$ un ensemble de transactions, une transaction étant un sous-ensemble d'*items* de I tel que $T \subseteq I$.
Un sous-ensemble $X = \{i_1, i_2, i_3\}$ non vide de T est appelé *itemset*s. Nous le notons I.
La longueur de I spécifiée par la valeur de k, correspond au nombre d'*items* contenus dans X, on le note : *k-itemsets*. Une règle d'association est un 2-*uple*t (X, Y) *itemsets* de T représentant une implication de la forme $X \rightarrow Y$ avec $X \subset I$, $Y \subset I$ et telle que $X \cap Y = \emptyset$.

Une règle d'association s'exprime généralement par : si $(x_1, x_2 \ldots x_n)$ alors $(y_1, y_2 \ldots y_n)$. Dans chaque partie de la règle (droite et gauche) se trouve une conjonction d'*items* au sens logique. La partie gauche $\{X\}$ est appelée prémisse ou condition de la règle et la partie droite $\{Y\}$, la conclusion.

BLANCHARD dit que ces règles signifient que si un enregistrement de la table d'une base de données vérifie la prémisse, alors il vérifie probablement la conclusion. La conclusion est en revanche complètement vérifiée pour une valeur de la mesure de confiance à 100%.

2.2.2 Propriété de fréquence des ensembles d'*items*

Un ensemble de taille *k* est appelé un *k-ensemble*, les « *k-ensembles* » fréquents constituent un semi-treillis. Tout *k-ensemble* fréquent est composé de (*k-1*) ensembles fréquents où encore, tous sous-ensembles d'un ensemble fréquent est fréquent. Un ensemble ne peut être fréquent si certains sous-ensembles ne le sont pas. Donc si {*A, B*} est un ensemble d'*items* fréquents, alors {*A*} et {*B*} sont aussi des ensembles d'*items* fréquents. Généralisons par tous sous-ensembles de (*k-1*) *items* d'un ensemble de (*k*) *items* fréquents est fréquent (anti-monotonie), et que les sur-ensembles d'ensembles non fréquents, sont non fréquents (monotonie).
Pour illustrer cette propriété, considérons par exemple les ensembles d'*items* ordonnés suivants:

Soit S = {(*A, B, C*) ; (*A, B, D*) ; (*A, C, D*) ; (*A, C, E*) ; (*B, C, D*)} avec *S* joint.

Par exemple (*A, C, D, E*) n'est pas un ensemble d'*items* fréquents puisque l'ensemble (*C, D, E*) $\notin S$. En revanche (*A, B, C, D*) $\in S$, cet ensemble d'*items* est donc fréquent. La propriété de fréquence des ensembles d'*items* permet de construire facilement les *itemsets* de façon incrémentale. Cependant nous avons vu que la taille de l'espace de recherche formée par les règles extraites croit de façon exponentielle en fonction du nombre d'attributs (*items*) qui décrivent les données. Effectivement, avec un seul ensemble de « *p* » *items* on peut construire 2^p-1 *itemsets* fréquents. Par conséquent, à partir de l'*itemset* {*A, B, C*}, six règles d'association potentiellement utiles apparaissent (TAB.1).

Prémisse	*Conclusion*
A	*B, C*
B	*A, C*
C	*A, B*
A, B	*C*
A, C	*B*
B, C	*A*

TAB.1 - Règles d'association

L'explosion combinatoire du nombre d'*itemsets* engendre en sortie des algorithmes un volume de règles très important. BLANCHARD (2005) dit que dans la pratique, « *les listes peuvent contenir plusieurs centaines de milliers de règles sans aucun ordre* ». Le bruit généré par ces volumes pour l'extraction de la connaissance va donc à l'encontre du principe d'intelligibilité et influence grandement la qualité de l'information perçue par l'utilisateur dans son processus de décision.

Afin de limiter l'espace de recherche dans les données, plusieurs solutions d'élagage et de classement ont été étudiées. Le but étant de d'éliminer les règles redondantes ou triviales et ne conserver que celles présentant un intérêt pour l'expert. Les indicateurs de qualité les plus connus pour « filtrer » les règles, sont le support et la confiance. Ces mesures sont basées sur la propriété de fréquence des *itemsets*. L'extraction des règles à l'aide de ce couple d'indices est généralement associée à des seuils de validité.

2.3 Mesures d'intérêt

2.3.1 Indices de support et de confiance

Si l'on considère la règle $X \rightarrow Y$, où X et Y sont des ensembles d'*items* de T.

- Le support d'une règle : $X \rightarrow Y$ définit le pourcentage exprimé σ % des enregistrements de T qui contiennent $X \cup Y$ par rapport au nombre total des enregistrements dans la base. Le support indique la portée de la règle.

- La confiance « c » définit le pourcentage des enregistrements de T qui contiennent $X \cup Y$ par rapport au nombre d'enregistrements contenant X, $c = \sigma_{X \rightarrow Y} \setminus \sigma_X$. L'indice de confiance donne la précision de la règle.

Si un *itemset* fréquent dont la valeur que procure son support est supérieur à un seuil de support minimal *minsup*, il sera qualifié de candidat. Les seuils de support et de confiance seront notés σ_{sp} et σ_{cf}. Un *itemset* est dit maximal s'il n'est un sous-ensemble d'aucun autre *itemset*. Il est dit fermé s'il n'existe aucun autre *itemset* qui le contienne et dont le support est supérieur ou égal à *minisup* ($\nexists Y / X \subset Y$ et $\sigma_Y \geq \sigma_X$). Nous noterons également n_A la cardinalité d'un ensemble A.

- Illustration du support et de la confiance

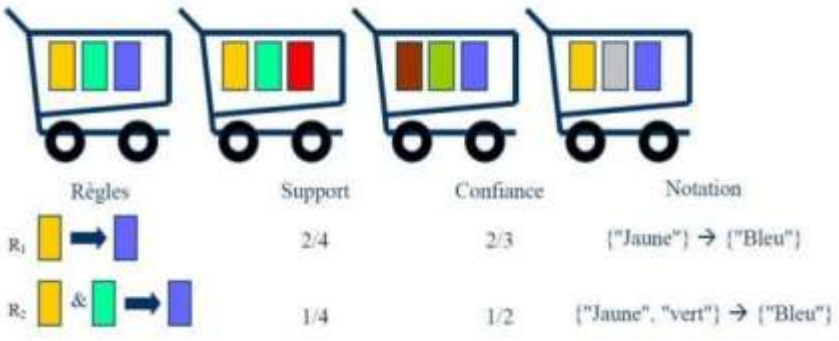


FIG.4 - Panier de la ménagère.

Nous détaillons les deux règles R_1 et R_2 suivantes et le calcul de leurs indices respectifs de support et de confiance :

- R_1 : {jaune} → {bleu}

  r_1.support = P (*jaune, bleu*) = $\frac{n\,(jaune,bleu)}{n\,(transactions)}$ = 50%.

  r_1.confiance = P (*jaune, bleu* / *jaune*) = $\frac{n\,(jaune,bleu)}{n\,(jaune)}$ = ou 66%.

- R_2: {jaune ∧ vert} → {bleu}.

  r_2.support = P (*jaune, vert, bleu*) = $\frac{n\,(jaune,\ vert,\ bleu)}{n\,(transactions)}$ = 25%.

  r_2.confiance = P (*jaune, vert, bleu* / *jaune, vert*) = $\frac{n\,(jaune,\ vert,\ bleu)}{n\,(jaune,\ vert)}$ = 50%.

L'attribution de seuils (*minsup, minconf*) à ces deux indices de qualité limite l'explosion combinatoire des *itemsets* fréquents et donc du nombre de règles. Ces mesures « historiques » possèdent des vertus algorithmiques accélératrices et se retrouvent dans la plupart des algorithmes d'extraction.

Une fois encore dans la pratique, le volume de règles en sortie des algorithmes reste élevé et prohibitif. De nombreuses règles sont redondantes ou sans intérêt. Définir les seuils pour les mesures de qualité reste discutable. En effet, un seuil élevé pour l'indice de support peut entrainer la discrimination de règles qui potentiellement pourraient représenter de la connaissance pour l'expert. En terme de connaissance, même un événement rare peut requérir de l'importance.

De nombreuses mesures statistiques associées aux règles d'association existent dans la littérature, COUTURIER O. (2005). Elles permettent d'enrichir la description des règles et d'écarter les moins intéressantes au profit des plus pertinentes. L'utilisation de ces mesures peut se faire au sens d'une métrique donnée et sous différents points de vue.

Lorsqu'elles sont associées à des seuils de qualité, ces mesures limitent une partie des règles qui ne correspondent pas au contexte de l'analyse. Elles sont classées suivant deux catégories en fonction de leur orientation : les mesures subjectives (orientées utilisateur) et les mesures objectives (orientées données).

2.3.2 Mesures d'intérêt objectives

TAN et *al*. (2004) montrent que les mesures objectives tiennent compte de la structure des données et plus particulièrement des effectifs liés à la contingence des données. Ces mesures sont en générale de nature statistique et adaptées à la distribution des données, ALESSANDRI (2010).

Pour les présenter, nous allons considérer les attributs présents dans une règle *r* de la forme $A \rightarrow B$. Ces valeurs sont déterminées par la table de contingence de *r* (FIG.5).
De nombreux articles comparent l'intérêt de ces indices. TAN et *al*. (2004) en présentent vingt et un (TAB.2). La comparaison se base sur deux types de critères :

- Des critères théoriques et mathématiques qui établissent un jeu de caractéristiques attendues. BLANCHARD et *al*, les confrontent avec les propriétés mathématiques des indices,

- Des critères expérimentaux, qui consistent à appliquer ces indices sur des jeux de tests, éventuellement en utilisant des outils informatiques de comparaison de règles (ARVAL[11]).

	B	$\overline{B}$	
A	p_{ab}	$p_{a\bar{b}}$	p_a
$\overline{A}$	$p_{\bar{a}b}$	$p_{\bar{a}\bar{b}}$	$p_{\bar{a}}$
	p_b	$p_{\bar{b}}$	1

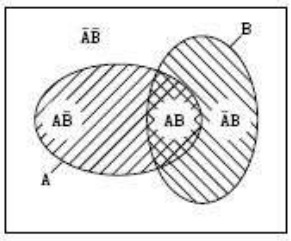


FIG.5 - Table de contingence de *r*

[11] ARVAL est un logiciel libre d'extraction de règles d'association. Il est particulièrement dédié au post-traitement et s'appuie sur des mesures objectives

#	Measure	Formula
1	ϕ-coefficient	$\frac{P(A,B)-P(A)P(B)}{\sqrt{P(A)P(B)(1-P(A))(1-P(B))}}$
2	Goodman-Kruskal's (λ)	$\frac{\sum_j \max_k P(A_j,B_k)+\sum_k \max_j P(A_j,B_k)-\max_j P(A_j)-max_k P(B_k)}{2-\max_j P(A_j)-\max_k P(B_k)}$
3	Odds ratio (α)	$\frac{P(A,B)P(\overline{A},\overline{B})}{P(A,\overline{B})P(\overline{A},B)}$
4	Yule's Q	$\frac{P(A,B)P(\overline{AB})-P(A,\overline{B})P(\overline{A},B)}{P(A,B)P(\overline{AB})+P(A,\overline{B})P(\overline{A},B)} = \frac{\alpha-1}{\alpha+1}$
5	Yule's Y	$\frac{\sqrt{P(A,B)P(\overline{AB})}-\sqrt{P(A,\overline{B})P(\overline{A},B)}}{\sqrt{P(A,B)P(\overline{AB})}+\sqrt{P(A,\overline{B})P(\overline{A},B)}} = \frac{\sqrt{\alpha}-1}{\sqrt{\alpha}+1}$
6	Kappa (κ)	$\frac{P(A,B)+P(\overline{A},\overline{B})-P(A)P(B)-P(\overline{A})P(\overline{B})}{1-P(A)P(B)-P(\overline{A})P(\overline{B})}$
7	Mutual Information (M)	$\frac{\sum_i \sum_j P(A_i,B_j)\log\frac{P(A_i,B_j)}{P(A_i)P(B_j)}}{\min(-\sum_i P(A_i)\log P(A_i), -\sum_j P(B_j)\log P(B_j))}$
8	J-Measure (J)	$\max\big(P(A,B)\log(\frac{P(B\|A)}{P(B)}) + P(A\overline{B})\log(\frac{P(\overline{B}\|A)}{P(\overline{B})}),$ $P(A,B)\log(\frac{P(A\|B)}{P(A)}) + P(\overline{A}B)\log(\frac{P(\overline{A}\|B)}{P(\overline{A})})\big)$
9	Gini index (G)	$\max\big(P(A)[P(B\|A)^2 + P(\overline{B}\|A)^2] + P(\overline{A})[P(B\|\overline{A})^2 + P(\overline{B}\|\overline{A})^2] - P(B)^2 - P(\overline{B})^2,$ $P(B)[P(A\|B)^2 + P(\overline{A}\|B)^2] + P(\overline{B})[P(A\|\overline{B})^2 + P(\overline{A}\|\overline{B})^2] - P(A)^2 - P(\overline{A})^2\big)$
10	Support (s)	$P(A,B)$
11	Confidence (c)	$\max(P(B\|A), P(A\|B))$
12	Laplace (L)	$\max\big(\frac{NP(A,B)+1}{NP(A)+2}, \frac{NP(A,B)+1}{NP(\overline{B})+2}\big)$
13	Conviction (V)	$\max\big(\frac{P(A)P(\overline{B})}{P(A\overline{B})}, \frac{P(B)P(\overline{A})}{P(B\overline{A})}\big)$
14	Interest (I)	$\frac{P(A,B)}{P(A)P(B)}$
15	cosine (IS)	$\frac{P(A,B)}{\sqrt{P(A)P(B)}}$
16	Piatetsky-Shapiro's (PS)	$P(A,B) - P(A)P(B)$
17	Certainty factor (F)	$\max\big(\frac{P(B\|A)-P(B)}{1-P(B)}, \frac{P(A\|B)-P(A)}{1-P(A)}\big)$
18	Added Value (AV)	$\max(P(B\|A) - P(B), P(A\|B) - P(A))$
19	Collective strength (S)	$\frac{P(A,B)+P(\overline{AB})}{P(A)P(B)+P(\overline{A})P(\overline{B})} \times \frac{1-P(A)P(B)-P(\overline{A})P(\overline{B})}{1-P(A,B)-P(\overline{AB})}$
20	Jaccard (ζ)	$\frac{P(A,B)}{P(A)+P(B)-P(A,B)}$
21	Klosgen (K)	$\sqrt{P(A,B)}\max(P(B\|A) - P(B), P(A\|B) - P(A))$

TAB.2 - Mesures objectives de règles d'association [Tan et al., 2004]

Les **critères théoriques et mathématiques** permettent de répondre aux problèmes liés à la différence des structures de données. Ils s'appuient sur huit propriétés pour une mesure de *M*,

Initialement, PIATETSKY-SHAPIRO (1991) propose trois principes applicables à toutes les mesures de *M* :

- P1 : $M_{A\to B} = 0$ si *A* et *B* sont statistiquement indépendants, $P(A, B) = P(A)\,P(B)$,
- P2 : $M_{A\to B}$ croit de façon monotone avec $P(A, B)$ quand $P(A)$ et $P(B)$ ne varient pas,
- P2 : $M_{A\to B}$ décroit de façon monotone si $P(A)$ ou $P(B)$ décroit, $P(A, B)$, et respectivement.

TAN et *al*. (2004) définissent cinq propriétés basées sur la table de contingence de *r* (FIG.5) :

- O1 : symétrie par permutation de variables,

- O2 : invariance (*M* reste identique) vis à vis des changements d'échelle sur chaque rang ou colonne,

- O3 : antisymétrie (*M* devient –*M*) par permutation des rangs et des colonnes,

- O4 : invariance par inversion, *M* reste identique si l'ensemble lignes/colonnes est inversé,

- O5 : invariance nulle. La mesure *M* n'a aucune relation avec le nombre d'enregistrements qui ne contiennent pas *A* et *B*.

À la différence des principes de Piatetsky-Shapiro, ces propriétés n'apportent pas d'indication directe sur les données analysées mais peuvent être utilisées pour classer les mesures en différents groupes.

- La propriété O1 déclare que les règles $A \rightarrow B$ et $B \rightarrow A$ peuvent avoir le même niveau d'intérêt, pour de nombreuses applications cette propriété n'est pas complètement vérifiée. En effet, la confiance représente la probabilité de la conclusion par rapport à la prémisse et non l'inverse. Cette mesure est donc asymétrique. Pour introduire de la symétrie dans les mesures d'intérêt. TAN et *al*. (2004) proposent de transformer les mesures asymétriques en mesures symétriques par l'utilisation du maximum de la valeur de M ($A \rightarrow B$) et M ($B \rightarrow A$). La symétrie d'une mesure de confiance est alors définie en considérant *max* (P($B|A$), P($A|B$)).

- La propriété O2 exige l'invariance dans le changement d'échelle entre les colonnes et les lignes.

- La propriété O3 indique que $M_{A\rightarrow B} = - M_{A\rightarrow \bar{B}} = - M_{\bar{A}\rightarrow B}$ .Cette propriété signifie que la mesure peut identifier des corrélations tant positives que négatives.

- La propriété O4 indique que $M_{A\rightarrow B} = M_{\bar{A}\rightarrow \bar{B}}$, O3 est alors un cas particulier de O4 parce que si la permutation des lignes/colonnes provoque un changement de signe, le signe est à nouveau changé par la permutation des colonnes/lignes. Le résultat global de la permutation de l'ensemble lignes/colonnes restera donc inchangé.

- La propriété O5 signifie que la mesure tient compte des enregistrements contenant *A*, *B* ou les deux. Bien que cette propriété puisse être applicable à l'indice de confiance, elle n'est pas vérifiée pour le support.

TAN et *al*. (2004) présentent un échantillon sur une population d'étudiants où la proportion d'hommes et de femmes par niveau de diplôme obtenu est identique mais où les distributions hommes/femmes sont différentes. L'indice GINI par exemple, fournit sur cet échantillon des résultats divergents, ALESSANDRI (2010).

La propriété d'invariance O2 pour les changements d'échelle sur un rang ou sur une colonne n'est donc pas vérifiée et des indices réciproques qui pourraient satisfaire cette propriété peuvent se révéler inappropriés pour d'autres circonstances.
Finalement, la comparaison des indicateurs fait apparaître qu'aucun d'entre eux ne respectent l'ensemble des critères proposés. Il est alors nécessaire de sélectionner l'un ou l'autre, en fonction des résultats attendus et des conditions de la mesure. Cependant, choisir une mesure par rapport à une autre dans le but de découvrir de la connaissance, prend un caractère de recherche empirique. En effet, le choix retenu influe directement sur le jeu de règles découvertes et l'information produite par l'utilisation d'une mesure peut être contredite par l'information d'une mesure différente qui serait fournie.

Pour les critères expérimentaux, TAN et *al*. (2004) applique un panel de vingt et un indicateurs à dix tables de contingence (notées d'E1 à E10). Ces tables vérifient la conformité des analyses théoriques puisque sur une même table, les résultats fournis par les indices divergent (ALESSANDRI (2010).

BLANCHARD et *al*., (2005) proposent une autre classification différente en montrant que la qualité objective des règles peut être évaluée suivant deux mesures supplémentaires : une mesure de déviation par rapport à l'indépendance et une mesure de déviation par rapport à l'équilibre (TAB.3).

Ces mesures correspondent à des écarts existants entre les nombres d'exemples et de contre-exemples. BLANCHARD dit que l'écart à l'équilibre est un constat absolu alors que l'écart à l'indépendance est une comparaison relative à une situation attendue (caractérisé par n_b).

Cette classification tient également compte des variations de l'indice en fonction de la cardinalité de l'ensemble des transactions.

- L'écart à l'indépendance est donné quand la prémisse *a* et la conclusion *b* sont indépendants ($n.n_b = n_a.n_b$),

- L'écart à l'équilibre est donné quand les exemples et les contre-exemples sont à l'équilibre ($n_{ab} = n_{a\bar{b}}$),

- Les mesures descriptives restent inchangées quelque soit la cardinalité de l'ensemble des transactions,

- Les mesures statistiques varient avec cette cardinalité.

	Measures of deviation from equilibrium	Measures of deviation from independence
Descriptive measures	– confidence, – Sebag et Schoenauer index, – example and counter-example ratio, – Ganascia index, – *moindre-contradiction*, – inclusion index...	– correlation coefficient, – lift, – Loevinger index, – conviction, – J-measure, – *TIC*, – odds ratio, – *multiplicateur de cote*...
Statistical measures		– implication intensity, – implication index, – likelihood linkage index, – oriented contribution to χ^2, – rule-interest...

TAB.3 - Classification des mesures d'intérêt objectives BLANCHARD et *al*., (2005)

Cette classification fait apparaître que les mesures d'écart à l'indépendance et les mesures d'écart à l'équilibre sont complémentaires. Ces deux mesures tiennent compte de l'interprétation individuelle de la notion de règles contrairement aux sens logique de l'implication existante entre la prémisse et la conclusion. Une règle d'association offre alors une lecture ouverte. BLANCHARD (2005) montre que la règle $(A \rightarrow B)$ peut être lue comme une description où les deux affirmations sont liées et se produisent généralement ensemble. On peut donc faire le choix d'ignorer les cas pour lesquels $A = 0$ et $B = 0$ ou au contraire (*dans la causalité* : *fumer → cancer*), de les considérer comme des exemples. On privilégiera l'écart à l'équilibre dans le premier cas. Pour le deuxième cas on s'intéressera plus particulièrement à la différence de probabilité d'apparition de la conclusion par rapport à la présence ou non de la prémisse. Ce dernier cas relève alors d'une mesure d'écart à l'indépendance.

Quelques mesures d'intérêt objectives :

- Le Lift

Le lift est une mesure d'écart à l'indépendance, cet indice donne une évaluation de la dépendance existante entre la prémisse et la conclusion. Un lift de faible valeur signifie que les *items* ne sont que faiblement corrélés entre eux. Plus la valeur du lift sera élevée et plus les *items* seront corrélés. Le lift est donné par la formule suivante :

$$Lift_{A \rightarrow B} = \frac{P(B|A)}{P(B)} \text{ ou encore } \frac{P(AB)}{P(A)\,P(B)}$$

- Le gain informationnel

Le gain informationnel est une mesure basée sur la théorie de l'information. La théorie de l'information de Shannon est une théorie probabiliste qui vise à quantifier la notion de contenu en information. En fonction de l'ensemble de données, l'information présente un caractère essentiellement aléatoire, un événement aléatoire est par définition, incertain. Une mesure clé de cette incertitude est connue sous le nom d'entropie. Intuitivement, l'entropie

quantifie l'incertitude de la valeur pouvant être prise lorsque nous sommes en présence d'une variable aléatoire. Objectivement, plus un événement est incertain et plus sa réalisation apporte de l'information mais aussi, plus cet événement est incertain et plus sa prédiction nécessite de l'information.

Pour une règle de la forme ($A \rightarrow B$), le gain informationnel que donne un attribut de A à l'égard d'un attribut de la classe de B est la réduction de l'incertitude sur la valeur de B, $I(B, A)$. L'incertitude sur la valeur de B est mesurée par l'entropie $H(B)$. L'incertitude sur la valeur de B lorsque la valeur de A est donnée par l'entropie conditionnelle de B est notée $H(B|A)$. Le gain informationnel pour une telle règle s'exprime alors par :

$$H(B|A) = log\ (P(AB) / P(A)\ P(B)).$$

2.3.3 Mesures d'intérêt subjectives

La sélection de règles d'association par des moyens statistiques ou structurels se heurte à deux problèmes principaux :

- faire face au grand volume de règles.

- L'élimination de règles intéressantes par des critères trop restrictifs, en particulier dans le choix de la valeur du seuil pour le support.

La découverte de règles intéressantes peut être induite par l'expertise de l'utilisateur. Cette notion fait l'objet des mesures subjectives. Elles offrent par des méthodes supervisées, des moyens d'introduire dans le processus de fouille de données, les savoirs et les questionnements des utilisateurs ALESSANDRI M., (2010).

LIU et *al*, (1997) proposent deux critères subjectifs, le premier s'appuie sur le caractère inattendu d'une règle (*unexpectedness*), le second sur son actionnabilité (*actionability*).

- L'inattendu exprime le fait qu'une règle est surprenante et qu'elle intéresse l'utilisateur par sa nouveauté.

- L'actionnabilité évalue l'aptitude d'une règle à être applicative pour une action précise et utile. La notion d'actionnabilité est utilisée plus particulièrement dans le cadre d'actions commerciales ou encore pour capitaliser les données d'un processus en termes de performances ou de qualité mais ne rentre pas dans le cadre de cette étude.

LIU et *al*, (1997) fournissent également des outils qui permettent d'exploiter ces deux critères par un langage de description sur les impressions générales de l'expert. Ce langage est associé à une méthode algorithmique pour classer les règles découvertes suivant des critères conforment aux impressions générales :

- Règles conformes (à un critère d'impression général défini),
- Règles dont les conclusions sont inattendues,
- Règles dont les conditions sont inattendues.

2.4 Algorithmes d'extraction

2.4.1 Algorithmes exhaustifs

Ces algorithmes effectuent tous la même tâche déterministe. BLANCHARD (2005) dit qu'à partir d'un seuil minimal de support et un seuil minimal de confiance, ils produisent un ensemble exhaustif de toutes les règles qui possèdent des indices de support et de confiance supérieurs aux seuils. A partir des *itemsets* et de leur propriété de fréquence (anti-monotonie du support), l'extraction des règles d'association s'organise alors autour de deux processus. Le premier consiste à trouver tous les *itemsets* fréquents candidats au seuil σsp, le second à partir de ces *itemsets*, de générer toutes les règles d'association valides au seuil σcf. Cette méthode est mise en œuvre par les algorithmes de type « *Apriori* » qui sont des références incontournables pour l'extraction de règles d'association. Nous ferons donc à ce titre, le détail des deux processus qu'ils mettent en œuvre. Nous détaillerons également l'algorithme CHARM qui effectue une recherche des *itemsets* fermés, une alternative intéressante pour faire face au volume de règles d'association extraites.

- Algorithme *Apriori*.

On considère un ensemble T de transactions t, L_k, un ensemble constitué de sous-ensembles d'*items* fréquents de longueur k et C_k un ensemble constitué des sous-ensembles d'*items* candidats de longueur k avec $L_k \subset C_k$.

Déterminer tous les ensembles fréquents de T nécessite au préalable de calculer le support de tous les *itemsets* et de tous les sous-ensembles susceptibles d'être fréquents. Notons pour cela, que la donnée C_k (ainsi que L_k) est un ensemble d'enregistrements contenant deux champs :

- le champ *itemset* un sous-ensemble d'*items*,

- le champ *count* pour la fréquence de l'*itemset* dans T.

Décomposition de l'algorithme *Apriori*:

1. Extraction des itemsets fréquents (TAB.4):

- Recherche et énumération des *1- ensembles* fréquents,

- Procédure *apriori-générer* : Elle est constituée de deux phases, la première phase génère tous les candidats possibles de longueur k à partir de l'ensemble L_{k-1}. La deuxième phase efface de C_k les éléments qui ne vérifient pas la propriété des sous-ensembles fréquents (élagage de C_k).

- Procédure *sous-ensembles* : l'algorithme calcule le sous ensemble $C_t \subseteq C_k$ qui correspond à des sous-ensembles présents dans les transactions de T.

Entrées : BD : base de données,
σ_{sp} : seuil de support minima,
Sortie : L, ensemble de couples (*I, sp (I)*) où *I* est un *itemset* et *sp(I)* son support.

L_1 = {fréquent 1-ensemble}
Calculer L_1

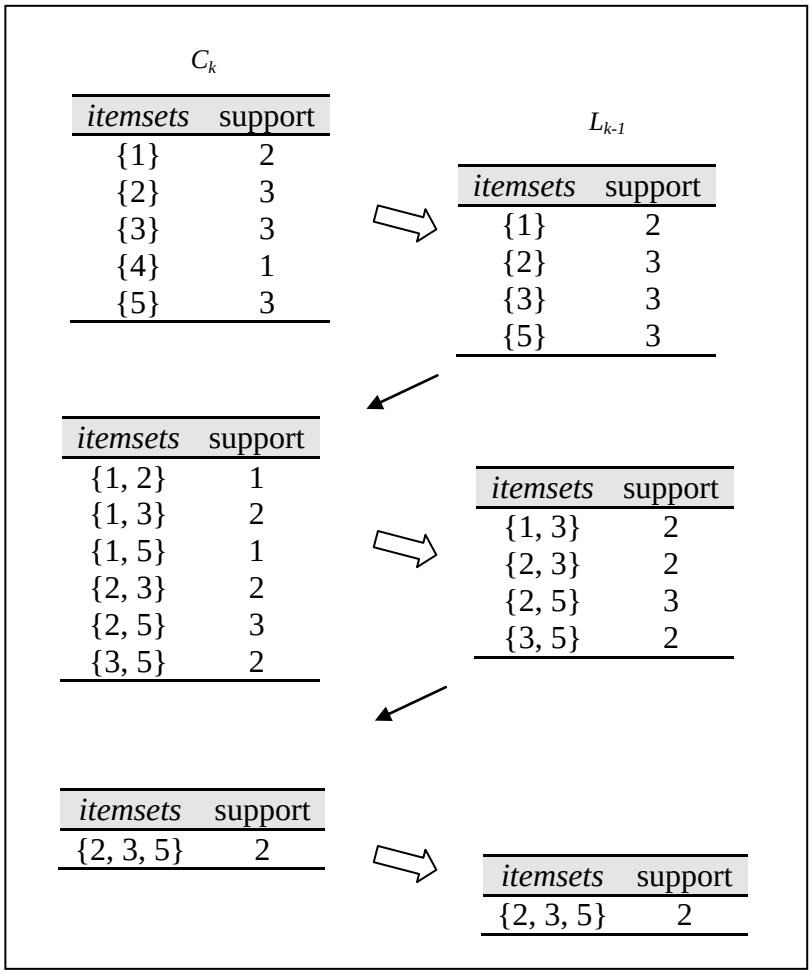
1. $k \leftarrow 2$
2. **tant que** $L_{k-1} \neq \emptyset$ **faire**
3. $Ck \leftarrow apriori\_gen\ (L_{k-1})$; // Elagage de C_k
4. **tant que** $t \in T$ **faire**
5. C_t = *sous-ensemble* (C_k ; *t*) ;
6. **tant que** $c \in C_t$ **faire**
7. c.count++ ;
8. **fin de tant que**
9. **fin de tant que**
10. $L_k \leftarrow \{c \in C_k \mid c.count \geq \sigma_{sp}\}$ // Les candidats sont filtrés
11. $k \leftarrow k+1$
12. **fin de tant que**
13. $L = \cup_k L_k$

TAB.4 - Extraction des *itemsets* fréquents dans *Apriori*

OID	*itemsets*
100	1, 3, 4
200	2, 3, 5
300	1, 2, 3, 5
400	2, 5

TAB.5 - Table des données

Les données de notre modèle (TAB.5) sont présentées en entrée de l'algorithme d'extraction. Nous détaillons (TAB.6) la méthode pour extraire les *itemsets* fréquents. L'algorithme procède à un élagage successif des *itemsets* de l'ensemble C_k qui ne vérifient pas la propriété de fréquence L_{k-1}. Les *itemsets* fréquents (partie droite) sont présentés en entrée de l'algorithme pour la validation des règles.



TAB.6 - Décomposition des sous-ensembles C_k et L_{k-1}

2. Génération des règles.

A partir d'un *itemset* fréquent *I*, l'algorithme construit toutes les règles de la forme $x \rightarrow y$ où x et y, sont deux sous-*itemsets* de *I* qui ne possèdent pas d'*items* en commun et qui redonnent *I* par conjonction : $x \wedge y = I$, BLANCHARD J., (2005). La confiance d'une telle règle est calculée de la manière suivante :

$$\text{Conf}\ (x \rightarrow y) = \frac{sp(I)}{sp\ (x)}$$

Entrées : Lk, ensembles d'*itemsets I* fréquents
σ_{cf}, seuil de confiance (*conf*) minima,
Sortie : R, ensemble de régles d'association

R = Ø
1. $k \leftarrow 2$
2. **tant que** $L_{k-1} \neq \emptyset$ **faire**
3. **tant que** *sous-ensemble* $S \neq \emptyset$ **de** L_k **faire**
4. $conf\,(S \rightarrow L_k - S) = sp\,(I) / sp\,(S)$
5. **si** $conf \geq \sigma_{cf}$
6. $r = S \rightarrow (L_k - S)$
7. $R = R \cup \{r\}$
8. **fin de si**
9. **fin de tant que**
10. $k \leftarrow k+1$
11. **fin de tant que**
12. **retourne** R

TAB.7 - Validation des règles

Au final, l'algorithme fournit l'ensemble des *itemsets* fréquents et les règles validées par le seuil de confiance. L'indice de support pour chaque *itemset* est conservé et sera utilisé pour le calcul des différentes mesures d'intérêt qui enrichissent les règles extraites.

- Algorithme d'extraction : CHARM

Zaki *et al,* (2002) utilise une structure arborescente mixte basée sur la correspondance de Galois pour rechercher de façon efficace et exhaustive l'ensemble des *itemsets* fermés. L'originalité de cet algorithme réside dans le fait qu'il privilégie l'exploration en profondeur dans l'espace de recherche contrairement à *Apriori* qui est un algorithme de parcours en largeur. L'idée étant d'exploiter la maximalité d'un *itemset* fermé.

Nous avons vu § *2.2.3* qu'un *itemset* fermé dans un ensemble d'objets, n'est pas inclut dans un autre *itemset*. L'algorithme CHARM explore donc simultanément l'espace de recherche des *itemsets* et celui des identificateurs des transactions notés *tidsets* fermés dans une structure appelée *IT-tree*. Cette méthode de recherche hybride évite l'exploration des nœuds inutiles. Sa représentation verticale appelée *difset* améliore l'efficacité des calculs. Le fonctionnement de l'algorithme CHARM est décrit par la trace (FIG.6) que nous détaillons ci-dessous.

Le symbole ($\neq$) indique une non inclusion ($\supset \subseteq$) et [X] est une liste des fils de l'*itemset* X.

- On commence par [Ø]= {A x 135, B x 2345, C x 1235, D x 1, E x 2345}

- Recherche de [A] : on prend le premier élément, A x 135 que l'on combine aux éléments du père ici [Ø] soit {A B C E} :

- AB x 35 comme e(A) $\neq$ e(B), AB x 35 est inséré dans [A] (fils de A)
- AC x 35 comme e(A) $\subset$ e(C) alors A est remplacé par AC

- ACE x 35 comme e(AC) ≠ e(E) alors ACE x 35 est ajouté à AC
- [AC] = {AB x 35, ACE x 35}

o Recherche de [AB] en partant de [AC] (profondeur)

- AB x 35 avec ACE x 35, comme e(AB) = e(ACE) alors AB est remplacé par ABCE.

o Recherche de [B]

- Comme e(B) ≠ e(C), BC x 235 est ajouté
- Comme e(B)=e(E), B et E sont combinés et E est enlevé de [Ø]
- B est remplacé par BE
- C ne peut être étendu et est donc inséré au résultat.

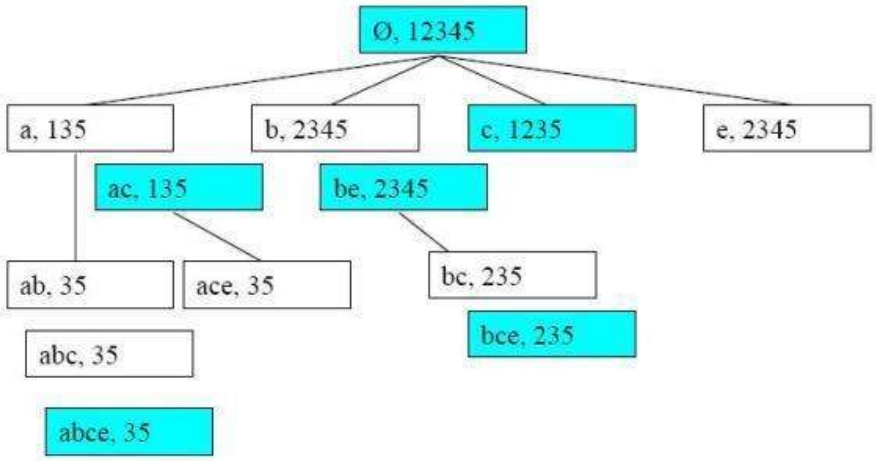


FIG.6 - Trace d'exécution de CHARM

2.4.2 Algorithmes à contraintes.

Les algorithmes à contraintes sont pour la plupart des généralisations d'*Apriori*. L'étape de génération de règles reste identique aux algorithmes exhaustifs. Cet algorithme différent par rapport à *Apriori*, dans le sens où ils sont optimisés pour une classe de contraintes.
Les deux classes principales sont les contraintes anti-monotones et monotones. Nous avons déjà présenté une application de la propriété d'anti-monotonie du support sur les *itemsets* fréquents (voir § 2.2.2). Les contraintes monotones sont exploitées lors de la génération des *itemsets* candidats. Cependant, afin d'éviter de diminuer l'efficacité de l'élagage réalisé par les contraintes d'anti-monotonie, l'utilisation des contraintes monotones nécessite de disposer de fonctions syntaxiques qui ne sont pas toujours disponibles pour l'énumération et la génération des *itemsets*. L'utilisation la plus efficace des contraintes monotones dans un processus d'extraction se trouve être à la phase de post-traitement de règles, c'est-à-dire, après l'extraction des *itemsets* BLANCHARD, (2005).

2.5 Post-traitement en ECD

En sortie des algorithmes d'extraction, les ensembles de règles d'association sont de simples listes textuelles. Chaque règle est constituée d'une conjonction d'*items* pour la prémisse et d'une conjonction d'*items* pour la conclusion. Les indices de qualité tels que le support et la confiance sont les plus utilisés. Pour faire face aux volumes de règles en phase de post traitement, trois voies ont été explorées. La première consiste par exemple, à utiliser une métrique spécifique pour filtrer et hiérarchiser les règles extraites ou encore de progresser par étapes successifs par le biais de différents résumés de règles. L'utilisateur converge alors vers les règles qui l'intéressent, en ajustant les seuils des mesures de qualité ou en spécifiant des contraintes. Nous avons vu § *2.2.7*, qu'un filtrage restrictif sur l'indice de support écarte les règles peu fréquentes et ces règles peuvent néanmoins comportées de l'information intéressante. Une deuxième approche vise à assister l'utilisateur dans son exploration en lui proposant des outils interactifs (navigateurs de règles, langages de requêtes). Enfin, une troisième méthode, consiste à offrir des moyens de représentation graphiques pour visualiser les ensembles de règles, plutôt que de les considérer sous forme textuelle. La visualisation graphique fait appel à des considérations ergonomiques sur la perception humaine. BEN-SAID et *al*, (2010) disent que l'œil humain est capable d'analyser rapidement et de façon synthétique son environnement pour y reconnaitre des informations d'un intérêt particulier ou des irrégularités. Notre étude portera donc sur la représentation graphique de l'information et nous mettrons en œuvre les technologies 3D pour la visualisation des règles d'association.

2.5.1 Exploration textuelle

Plusieurs outils interactifs sous forme de listes textuelles ont été développés pour assister l'utilisateur dans la fouille de règles. LIU et *al*, présentent une méthode d'interaction à l'aide d'outils agissant sur les seuils des mesures de qualité et en exploitant les connaissances a priori de l'expert. Celui-ci exprime alors les relations potentielles dans les données et leur degré de précision. Dans Ma et *al*., (2000) l'utilisateur explore un résumé de l'ensemble des règles extraites. A partir des éléments sélectionnés dans ce résumé, l'utilisateur peut accéder aux règles correspondantes dans l'ensemble des données d'origine.

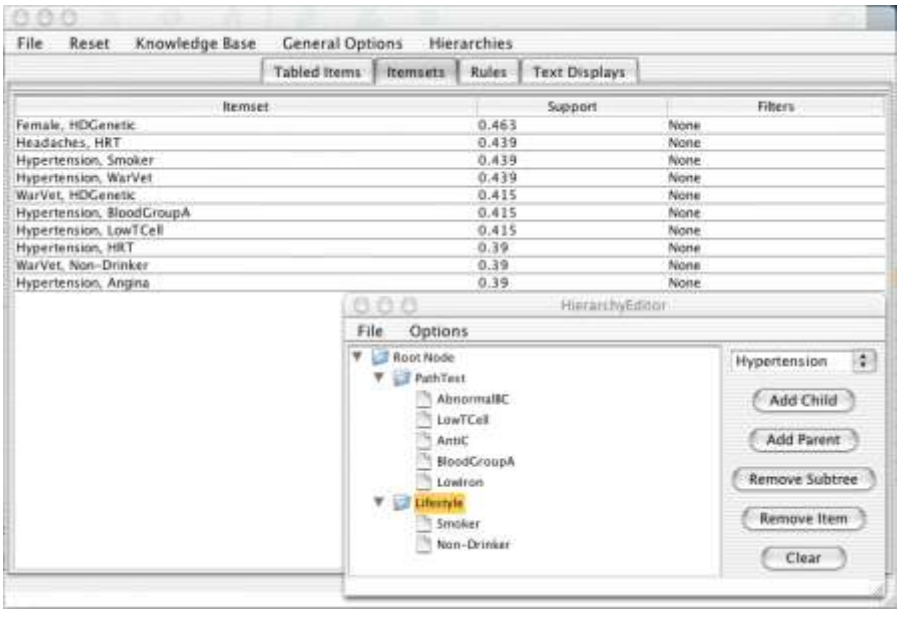


FIG.7 - L'explorateur de règles *IRSetNav*.

FULE et *al*, (2004) proposent un outil d'exploration textuelle de règles appelé *IRSetNav* (FIG.7). Il est doté de nombreuses fonctionnalités et filtre les règles par des contraintes syntaxiques plus ou moins générales en prenant en compte une taxonomie[12] des items. L'outil dispose de fonctionnalités pour programmer les indices de qualité, trier et filtrer les règles.

Nous retiendrons que le mode de représentation textuel implémenté par ces outils ne convient pas à l'exploration des règles d'association. Devant les volumes importants, l'interprétation des règles reste impossible. D'une façon générale ces outils sont donc inadaptés à la phase de post traitement d'un processus ECD.

2.5.2 Langage de requêtes

IMIELINSKI et MANNILA (1996) se penchent sur les langages de requêtes qui introduisent le concept de bases de données inductives. L'idée étant d'enrichir les Systèmes de Gestion de base de données en développant un langage de requêtes particulier pour la fouille de données. C'est-à-dire une généralisation de SQL qui permettrait de manipuler les données mais également et directement les connaissances extraites. Pour ce projet ambitieux de nombreux défis restent à relever puisqu'il est difficile, d'optimiser l'extraction des règles d'association pour des contraintes qui par essence, ne sont pas connues à l'avance.

2.5.3 Explorations graphiques

FAYYAD et *al*., (2001) disent que la visualisation peut être bénéfique à l'ECD. CARD et *al*., (1999) et POLENCO (2002) montrent que la visualisation est un moyen efficace d'introduire la subjectivité de l'utilisateur dans chaque étape du processus tout en amplifiant la cognition[13]. C'est-à-dire réduire le travail cognitif de l'utilisateur mais nécessaire pour accomplir certaines tâches. SILBERCSHATZ et TURKHILIN, (1996) disent que le processus ECD est hautement itératif et interactif, il requière donc l'implication de ce dernier. La visualisation est utilisée soit en tant que méthode d'extraction de données et dans ce cas nous parlons le plus souvent de *visual data mining*, soit en collaboration avec des algorithmes de fouille de règles ou celle-ci sont validées par l'expert dans cette phase de post traitement. L'approche fait intervenir la notion d'interaction associée à une stratégie de navigation dans un environnement complexe. L'utilisateur adopte alors une démarche empirique vers un but précis par le butinage, qui correspond à un besoin initialement mal exprimé. Il décide alors d'arrêter la procédure, lorsqu'il a obtenu satisfaction. AGGARWAL (2002) montre que ces algorithmes facilitent et accélèrent l'analyse des données, l'obtention des résultats intermédiaires et *in fine* l'extraction de la connaissance. Le système est construit de telle sorte que l'utilisateur puisse sélectionner lui-même des *itemsets* particuliers ainsi que les contraintes pour l'algorithme d'extraction. KUNTZ et, *al* (2006) posent un cadre méthodologique pour ce type d'approche.

[12] La taxonomie est la science des classifications

[13] La cognition regroupe les divers processus mentaux allant de l'analyse perceptive à l'appropriation dans des schémas et des concepts, par lesquels nous construisons une représentation aléatoire de la réalité à partir de nos perceptions, susceptible en particulier de nourrir nos raisonnements.

1. Caractéristiques de l'environnement, extraction des propriétés et opérations pertinentes,

2. Choix d'une représentation formelle pour l'espace de représentation et de navigation,

3. Implémentation informatique de cette représentation formelle : encodage et visualisation graphique,

4. Procédure de découverte de la connaissance : l'interactivité.

Lorsque la sélection s'effectue directement dans la représentation graphique par l'utilisateur, les modes d'interaction offerts par les objets qui la compose augmentent la compréhension et la perception de l'information.
Plusieurs méthodes graphiques pour représenter l'information ont été développées. Nous allons présenter les principales représentations graphiques que nous trouvons dans la littérature.

1. Représentation matricielle.

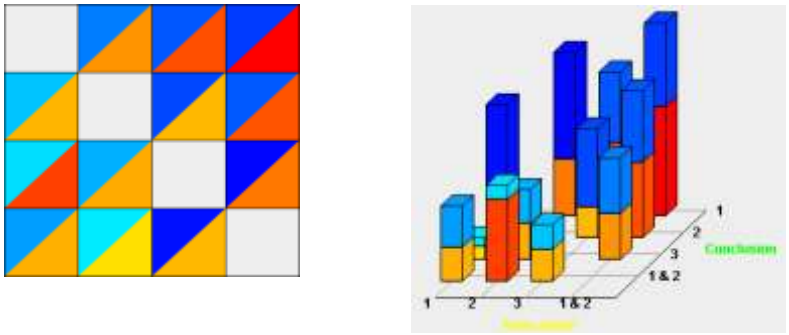


FIG.8 - Une matrice *item*-à-*item* de visualisation 2D et 3D dans *LARM*[14]

Une première méthode de visualisation de règles d'association est la représentation matricielle. Dans l'exemple de matrice item-à-item (FIG.8), chaque ligne correspond à un *item* en conclusion et chaque colonne à un *item* en prémisse. L'intersection des lignes et des colonnes symbolise les règles par des objets 2D ou 3D. Les caractéristiques graphiques des objets représentent les mesures de qualité (couleur ou dimension).

La techniques est améliorée par WRONG et *al*., (1999) qui proposent une matrice *item*-règles (FIG.9) permettant de gagner de l'espace et d'augmenter l'intelligibilité de la représentation par un encombrement plus faible de la matrice. La méthode impose cependant à l'utilisateur de faire lui-même la correspondance entre les règles et les mesures de qualité, en suivant la perspective.

[14] Acronyme *pour Large Association Rules Mining* développé dans (Couturier *et al*., 2005c).

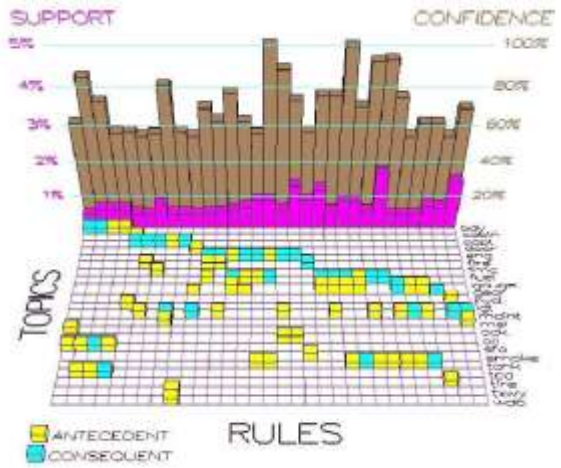


FIG.9 - Une matrice *item*-règles

2. Représentation par graphes

Une autre méthode de visualisation se présente sous forme d'un graphe orienté (FIG.10-*a*). Les nœuds et les arcs représentent respectivement les *items* et les règles. Les mesures de qualité sont encodées graphiquement par l'épaisseur ou la couleur de l'arc reliant les nœuds.
Pour des grands ensembles de règles, le graphe se surcharge rapidement de nœuds et d'arcs qui se croisent. Une solution dans LEHN., (2000) propose une représentation dynamique sous la forme d'un sous-graphe du treillis des *itemsets* (FIG.10-*b*). Les *items* représentés par les nœuds sont remplacés par d'autres *itemsets* de telle sorte qu'une règle de la forme : (*A*, *B*) → (*C*) est symbolisée par un arc entre les nœuds (*A*, *B*) et (*A*, *B*, *C*). L'utilisateur développe alors le graphe à sa guise en interagissant avec les nœuds. Le résultat est un graphe acyclique qui comporte plus de nœuds et moins de croisements d'arcs.

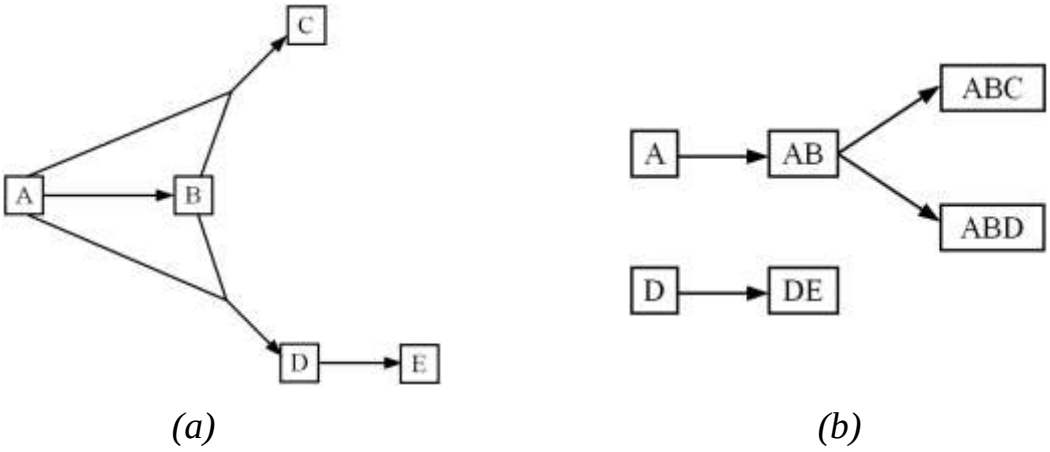


FIG.10 - Un graphe d'*items*(*a*), un graphe d'*itemsets*(*b*)

3. Comparaison des représentations matricielles et par graphes

Dans le mode de représentation matricielle, la matrice est rapidement limitée par le nombre de règles. L'amélioration apportée par une matrice *items*-règles demande à l'utilisateur des efforts mentaux et visuels qui croissent avec le nombre de règles et par son importance en diminue l'efficacité. La représentation par graphe a le mérite d'être plus intuitive mais admet deux principales limites :

1. l'usage du graphe fait implicitement apparaître les règles comme des relations transitives, une propriété qui ne s'applique pas aux règles d'association.

2. pour les indices, les mesures de qualité ne se propagent pas par transitivité BLANCHARD (2005).

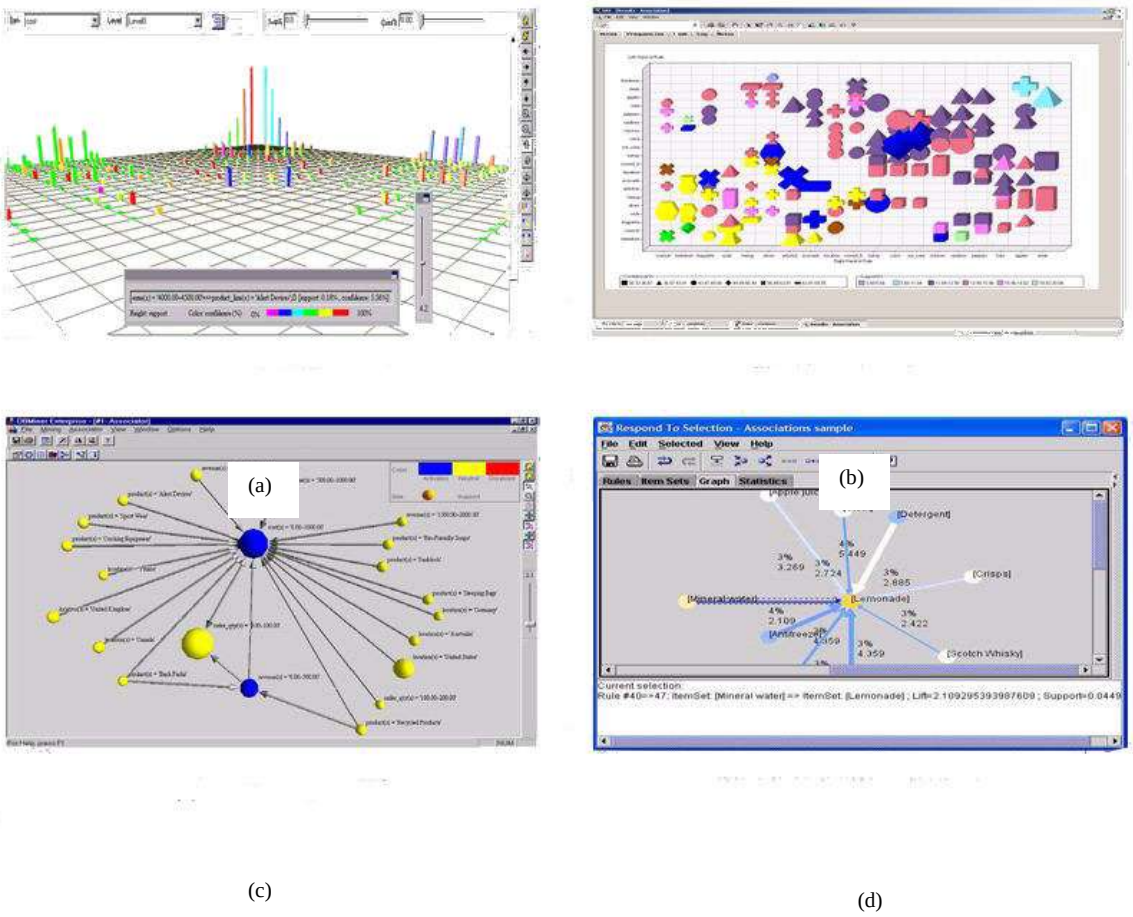


FIG.11 - Visualisation de règles par matrices (a : *DBMiner*, b : *Entreprise Miner*[15]) et graphes (c : *DBMiner*, d : *Intelligent Miner*[16]) BLANCHARD, (2005)

[15] Enterprise Miner. www.sas.com/technologies/analytics/datamining/miner/
[16] Intelligent Miner Visualization. www-3.ibm.com/software/data/iminer/visualization/ index.html

Les méthodes de représentation par graphes ou matricielles sont implémentées dans de nombreuses applications (FIG.11). Ces méthodes s'avèrent rapidement limitées devant les volumes importants de données qu'il est fréquent de traiter en ECD. Nous allons donc présenter les techniques utilisées en visualisation d'information dans un sens plus générale et définir ensuite un cadre d'étude applicatif à l'ECD en se basant sur les technologies graphiques les plus innovantes.

2.6 Visualisation de l'information

La visualisation de l'information est un domaine extrêmement vaste. En ECD, la visualisation des règles d'association reste une question ouverte. Les outils traditionnels, précurseurs en matière en fouille de données offrent pour la plupart une représentation 2D avec des interactions temps réel limitées. Depuis plus d'une vingtaine d'années, la puissance de calcul des ordinateurs et simultanément des cartes graphiques ont augmenté ostensiblement les possibilités dans le domaine de la simulation informatisée. Les capacités graphiques aujourd'hui, peuvent plonger l'utilisateur au cœur d'un monde totalement artificiel. L'utilisateur évolue alors dans un environnement virtuel 3D auquel nous pouvons associer les techniques d'interaction empruntées au domaine de la réalité virtuelle. D'une façon générale la réalité virtuelle simule les actions de la vie réelle. En ECD, les données à l'état brut n'ont ni forme, ni couleur, ni dimension. Le problème principal de la recherche dans ce domaine est alors d'imaginer de nouvelles métaphores pour interpréter graphiquement l'information. Toute la difficulté qui tient également de l'objectif, consiste à offrir par le biais de ces métaphores des clés d'interprétation pour observer des phénomènes et comprendre les tâches analytiques qu'elles peuvent supporter pour construire du sens.

CARD et *al*, (1999) définissent la visualisation d'informations comme l'utilisation de représentations informatiques interactives de données abstraites pour renforcer la cognition. Selon le type de données ou d'informations qu'on souhaite en retirer, il convient donc de réfléchir au choix de la visualisation à mettre en œuvre pour atteindre cet objectif. Les prochains paragraphes passeront en revue les différentes techniques de représentation graphiques existantes.

2.6.1 Modélisation des outils

Le modèle générique présenté dans CARD et *al*, (1999) est une suite de traitements interactifs pour passer des données en entrée à la visualisation en sortie (FIG.12). Les données d'entrée sont des ensembles d'entités décrites par les variables. Elles sont transformées puis encodées graphiquement. L'utilisateur conserve après encodage graphique des données, la possibilité d'effectuer de nouvelles transformations directement sur les vues.

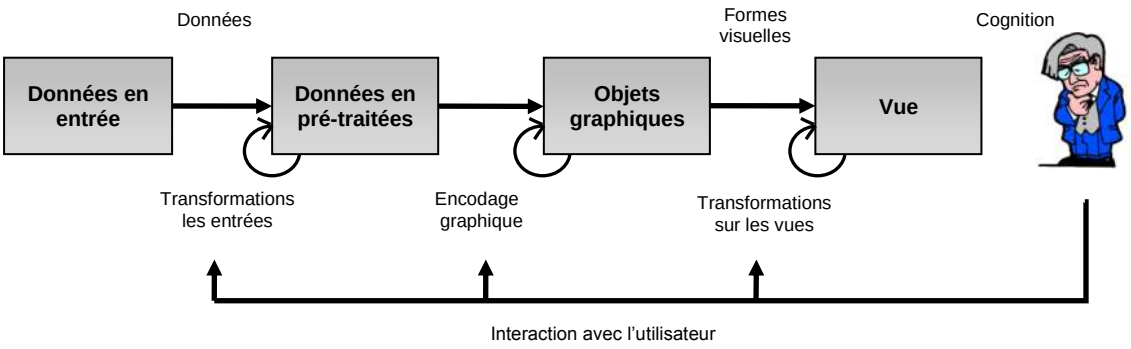


FIG.12 - Modèle générique pour la visualisation d'information CARD et *al*, (1999)

1. Les fonctionnalités de transformation sur les entrées effectuent les opérations de sélection, de formatage, de regroupement ou d'ordonnancement des modalités d'une variable, etc. Cette action peut être réalisée par des requêtes dynamiques (case à cocher[17], *sliders*[18]). On trouve également des techniques particulières pour améliorer ou ajouter des détails supplémentaires à la représentation graphique (infobulle, *pop-up, fisheye).*

2. L'encodage graphique est le cœur de la visualisation graphique. Les données sont réécrites sous forme d'objets graphiques. Chaque variable dans les données est associée à une composante graphique comme la couleur, la taille, la position, etc. Les objets graphiques peuvent être de une à trois dimension(s). L'évolution des objets dans le temps peut constituer une dimension supplémentaire et pour mettre en œuvre cette variable, une interface classique composée de nœuds et d'arcs permet de montrer les évolutions d'un graphe.

3. Les transformations sur les vues concernent la présentation des objets graphiques. La vue est soit en 2D, soit en 3D ou repose sur la métaphore du paysage d'informations dans un environnement virtuel interactif (FIG.13). Les techniques d'interaction les plus courantes affectent soit le contrôle du point de vue de façon exocentrique[19] ou egocentrique[20], soit utilisent des vues multiples (*overview + details*) ou intègrent les détails dans la vue globale (*focus + context*). Une technique classique de *focus + context* consiste à limiter graphiquement les détails d'une zone par effet de distorsion de l'image (FIG.14).

[17] Système de sélection visuel et interactif pour représenter des variables qualitatives.
[18]Un *slider* est un composant d'interface graphique permettant d'entrer une valeur numérique dans un programme en déplaçant un curseur sur une échelle graduée. Ce principe est utilisé en visualisation pour représenter des variables quantitatives.
[19] Le point de vue est fixe, la représentation subit des transformations (rotation, translation ou zoom)
[20] La représentation est fixe, le point de vue se déplace autour de la représentation.

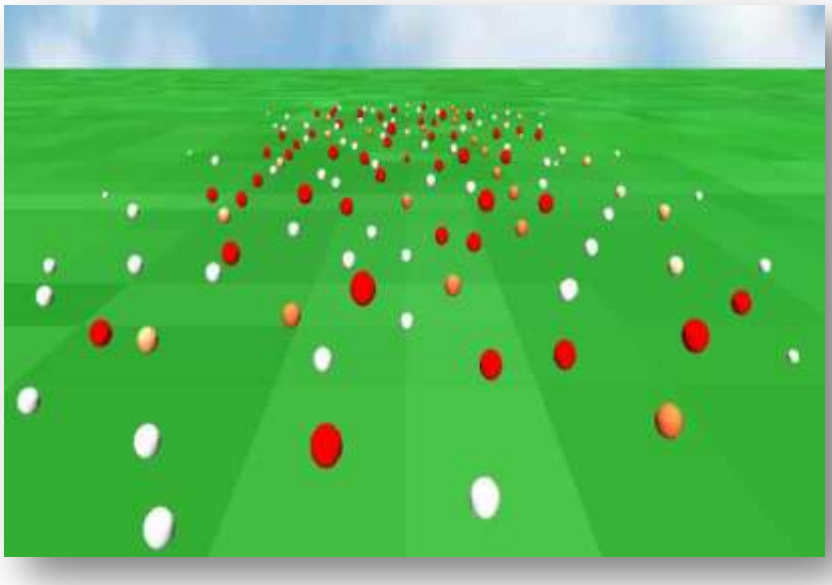


FIG.13 - Paysage d'informations, BLANCHARD (2005)

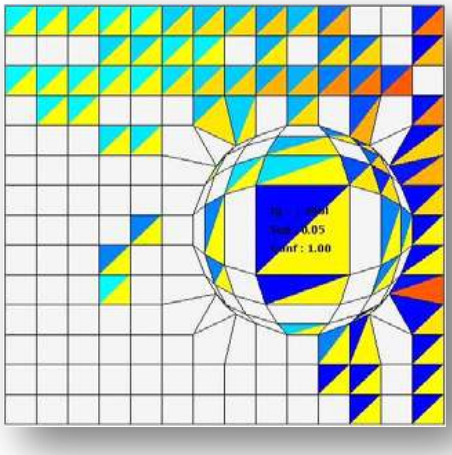


FIG.14 – *Fisheye*, CHEVRIN., et al (2005)

Les techniques peuvent être combinées comme par exemple le zoom sémantique qui transforme la vue et change les données suivant le niveau de détail demandé. La méthodologie présentée dans ce rapport s'efforce de parcourir le modèle sur ces trois composantes.

2.6.2 Sémiologie

La sémiologie[21] est la science des signes, le terme figure au *Littré*[22] pour définir en médecine, l'étude des symptômes présentés par les patients. Il fut repris et élargi pour l'étude des signes au sein de la vie sociale. Aujourd'hui, nous considérons que toute science qui étudie les signes est une sémiologie. Le terme est alors utilisé dans plusieurs disciplines. En représentation graphique, le géographe Jacques Bertin (1918-2010) est le premier à étudier un système de signes pour répondre aux besoins de la cartographie. Sa théorie, *la sémiologie graphique* [Bertin, 1967] énonce que le lecteur d'une carte perçoit six variations sensibles attachées aux symboles qui y figurent (FIG.15). Il les appelle variables visuelles ou « *composantes du système d'expression* ». Ces variables graphiques sont la position, la taille, la luminosité (valeur), la couleur, la texture (grain), l'orientation, et la forme.

Nous noterons que la position est la variable rétinienne prédominante dans une représentation graphique BERTIN (1967), CARD et *al*, (1999).

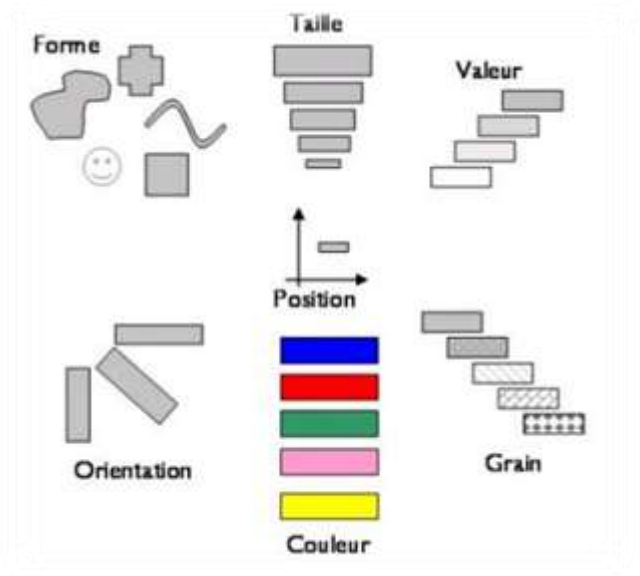


FIG.15 - Les variables rétiniennes de Bertin[23]

2.6.3 Représentations graphiques 2D *vs* 3D

Précisons que les modes de représentation graphiques en 2D ou en 3D ne font pas appel aux mêmes mécanismes cérébraux. Les représentations 2D n'induisent pas la même perception sur les données qu'en 3D et sont plus faciles à appréhender. Il est important de considérer cette différence afin de définir clairement l'objectif à atteindre et l'utilisation souhaitée pour faire le choix d'un mode de représentation le plus appropriée. La composante supplémentaire en 3D apporte, certes, une profondeur à l'infini et ouvre un champ de représentation beaucoup plus spacieux qu'en 2D limitée par la taille de l'écran. Cependant, il est difficile de se repérer dans une représentation 3D.

[21] Dictionnaire de Médecine, 1855. *Partie de la médecine qui traite des signes des maladies.*
[22] Dictionnaire normatif de la langue française du nom de son principal auteur Émile Littré (1801-1881)
[23] *http://www.knowledge-mapping.net/*

La perception de la profondeur n'est pas triviale, c'est-à-dire que la taille perçue des objets dépend étroitement de la position de ceux-ci et suivant la profondeur à laquelle ils se trouvent dans la scène. Notons également que le risque d'occultation entre les objets est une contrainte supplémentaire dans une représentation 3D. Néanmoins, les objets représentés en 3D admettent un nombre de caractéristiques graphiques supérieurs à la 2D et peuvent donc traduire un plus grand nombre d'informations. De plus, la navigation dans une scène 3D est intuitive. D'après WEGMAN et *al*, (2002), l'utilisation de systèmes de visualisation immersifs en réalité virtuelle comme un *visiocube* ou un visiocasque accentuent encore davantage le caractère pseudo-naturel de la navigation pour l'utilisateur. Le couplage des techniques de fouilles de données aux techniques de réalité virtuelle est un axe de recherche intéressant en visualisation de l'information.

2.7 Réalité Virtuelle

2.7.1 Définition, concept et enjeux

La réalité virtuelle (RV) est une simulation informatique interactive, immersive, visuelle, sonore et/ou haptique, d'environnements réels ou imaginaires. D'après FUCHS et *al*, (2001) la réalité virtuelle permet à l'utilisateur de s'extraire virtuellement du monde réel pour changer de temps, de lieu, et d'interaction. En se basant sur le concept de relation multidimensionnelle entre hommes et machines, la réalité virtuelle revêt déjà un agglomérat d'avantages. Elle donne l'accent à l'expérimentation. Elle est anticipatrice, globalisante et éveillante. Nous retiendrons de la littérature deux mots clés importants qui définissent les concepts de réalité virtuelle :

- l'immersion,
- l'interaction.

Son objectif est de rendre la machine transparente à l'utilisateur et de lui donner l'impression d'interagir avec l'application. Nous parlons alors de relation homme-application. L'utilisateur est en immersion pseudo-naturelle, c'est-à-dire qu'il agit sur le monde virtuel de la même façon qu'il agirait sur le monde réel. Mais cette notion est liée aux attentes des utilisateurs potentiels, compte tenu des moyens à mettre en œuvre pour simuler le plus fidèlement possible les actions du monde réel (pilotage d'un avion, acte chirurgical, etc.).

La RV trouve néanmoins des applications dans les domaines ou l'erreur n'est pas admise. En médecine on trouve sur *Medlin*e [24]plus de deux cents publications qui traitent de la RV appliquée à la chirurgie (scalpels intelligents à retour haptique, caméra tridimensionnelle, etc.). Une innovation récente propose un poignet artificiel et pourrait bien transformer le chirurgien de demain par ses performances supérieures à l'humain. Dans ce domaine, la réalité virtuelle est « dopée » par la réalité augmentée qui vise à compléter notre perception du monde réel. Notre étude ne rentre pas ce cadre.

Dans l'industrie, les constructeurs en automobile, aérospatial ou navale ont besoin de revoir le produit aux différentes étapes du processus de conception. Ils ont besoin notamment d'effectuer des tests sur la maquette numérique comme si, celle-ci, était réelle. L'objectif étant de réduire les coûts, en diminuant le nombre de prototypes physiques et de *facto*, de gagner du temps dans la mise en œuvre des produits (FIG.16).

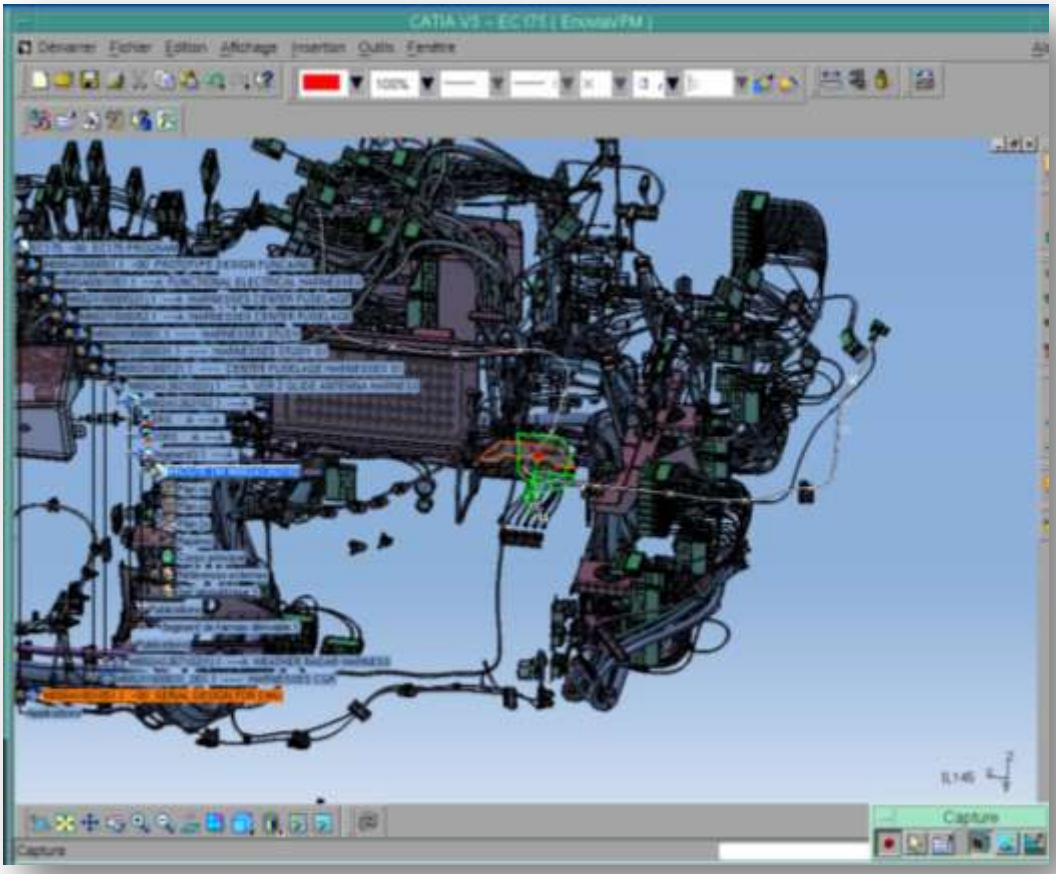


FIG.16 - Conception assistée par ordinateur (*CATIA*[25])

[24] MEDLINE est une base de données bibliographiques qui couvre tous les domaines médicaux de l'année 1966 à nos jours. http://www.nlm.nih.gov/databases/databases_medline.html

[25] Conception Assistée Tridimensionnelle Interactive Appliquée, logiciel créé à l'origine pour la conception d'aéronefs et propriété de Dassault systèmes.

En RV, les utilisateurs doivent agir avec les objets qui composent le monde virtuel. La notion de paradigme d'interaction est définie par certains auteurs pour désigner un ensemble de règles et de techniques accomplissant des tâches d'interactions au sein d'un environnement virtuel. La réalité virtuelle apporte donc un paradigme nouveau d'interaction pour la recherche de connaissances dans un processus ECD.

2.7.2 Apport de la réalité virtuelle à l'ECD

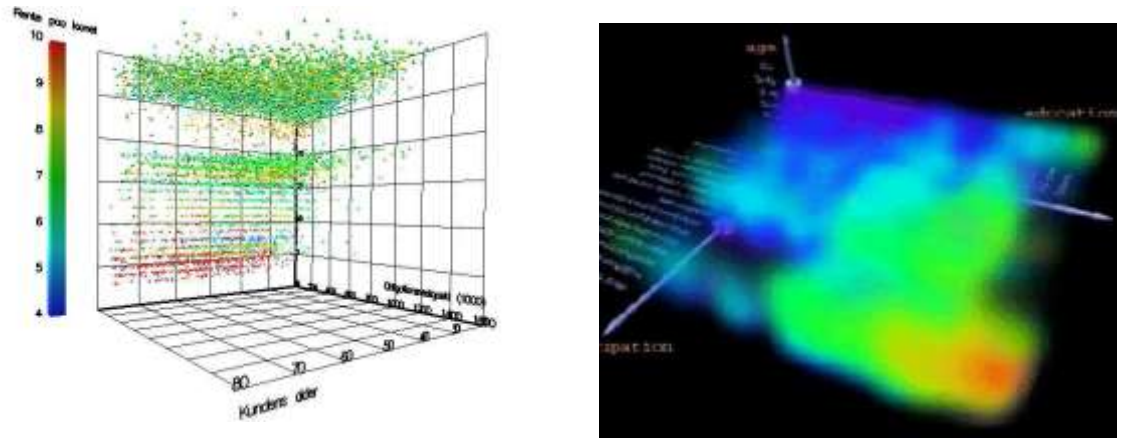
Contrairement à la grande majorité des applications de réalité virtuelle qui tendent à imiter la vie réelle, nous avons vu que les applications développées pour l'ECD se heurtent à la difficulté de donner du sens graphique à un concept abstrait que sont les mesures d'intérêt. Ces objets permettent d'interagir avec un monde de symboles qui correspond à une représentation intellectuelle et mentale « concrétisant » les concepts de données ou de connaissances via des métaphores FUCHS et *al*., (2001).

La métaphore doit traduire les termes d'un domaine particulier par des termes plus compréhensibles et familiers pour l'expert des données étudiées. C'est-à-dire interpréter graphiquement l'information dans la sémantique métier de l'utilisateur. Pour cela, il est possible dans un contexte de réalité virtuelle, d'imaginer une représentation et une interaction personnalisées d'une information complexe et volumineuse. L'introduction de la réalité virtuelle dans le processus ECD augmente plus encore les possibilités d'interaction et l'espace de visualisation pour les données à représenter.

L'interactivité forte et intuitive induites par les techniques de réalité virtuelle exploitent les capacités sensori-motrices humaines. Les applications en l'ECD basées sur ce principe font partie de ce que l'on a déjà nommé le *visual data mining*. Elles vont des logiciels de visualisation 3D interactifs pouvant s'exécuter sur un équipement informatique classique jusqu'aux systèmes immersifs en environnement virtuel nécessitant des interfaces spécifiques et des périphériques dédiés plus coûteux (bureau virtuel, visiocasque, gants de données).

2.7.3 Visualisation 3D de l'information

Une des représentations 3D les plus courantes en visualisation d'information est le nuage de points 3D (FIG.17). Le nuage de points indique le degré de corrélation entre deux ou plusieurs variables liées. Chaque unité représente un point dans le nuage. La 3D présente l'avantage par rapport à la 2D, d'offrir un rendu volumique. Le nuage de points est proposé dans de nombreux logiciels d'analyse de données.



(*a*) (*b*)

FIG.17 - Nuages de points, sans rendu volumique (*a*), avec rendu volumique (*b*)

La 3D et la réalité virtuelle trouvent également des applications dans la visualisation de graphes. Les arbres coniques (FIG.18-a) font partie des exemples les plus connus pour ce type de représentation. Il s'agit d'une structure hiérarchique d'arbres interactifs dessinés en 3D verticalement et horizontalement. Les nœuds enfant sont organisés de façon circulaire autour de leur nœud parent. Une action de l'utilisateur sur un nœud enfant entraine la rotation de toute sa hiérarchie au premier plan de la scène et permet ainsi de l'explorer horizontalement.

(*a*) (*b*)

FIG.18 - Visualisation d'arbres coniques (*a*) et métaphore botanique (*b*)

KLEIBERG et *al*, (2001) proposent une approche des arbres 3D radicalement différente avec une métaphore botanique (FIG.18*b*). La base de l'arbre représente les sommets des hiérarchies (*root*), les éléments sous catégorisés dérivent du tronc par les branches, les éléments finaux sont figurés par les feuilles et les ensemble d'éléments par les fruits.

MUNZNER et *al*, (2000) exploitent les propriétés des espaces hyperboliques[26] en généralisant les plans hyperboliques 3D (FIG.19). Les graphes dessinés dans ces espaces sont contenus dans des sphères selon l'approche *focus+context* où le centre de la sphère est magnifié et la périphérie peu détaillée.

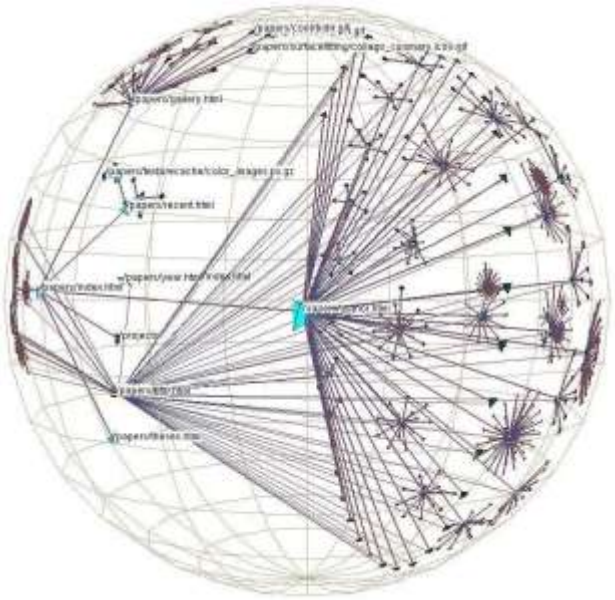


FIG.19 - Visualisation d'un arbre dans un espace hyperbolique (*focus + context*)

Les mondes virtuels constituent un courant important de la visualisation de l'information en 3D. L'apport de la réalité virtuelle avec les modes d'interaction qu'elle procure à l'utilisateur ouvre un champ d'exploration nouveau pour la recherche de connaissance dans les données. L'idée étant de représenter les données par des objets répartis dans un espace de grande taille et dans lequel l'utilisateur a la possibilité de naviguer par des primitives de navigation (contrôle du point de vue). Il peut également pointer les objets de la scène avec des périphériques classiques ou dédiés à la 3D. Les techniques de visualisation déjà présentées comme *focus + context* ou *overview + details*, sont des exemples d'interaction qu'il est intéressant, voire nécessaire, de mettre en œuvre dans une scène de réalité virtuelle.

Parmi les systèmes existants qui implémentent ces outils de visualisation pour l'exploration de données multi-dimensionnelles en visiosalle, nous trouvons le système TIDE développé par JOHNSON et LEIGH (2001). Ce système projette les données en nuages de points 3D. Il est fondé sur une architecture de travail collaborative et permet à plusieurs utilisateurs distants et matérialisés par le bais d'avatars d'explorer les données ensemble et de pouvoir échanger oralement sur celles-ci.

[26] Les plans hyperboliques pour la visualisation présentent l'intérêt d'afficher des entités non bornés (droite) dans une surface de visualisation bornée.

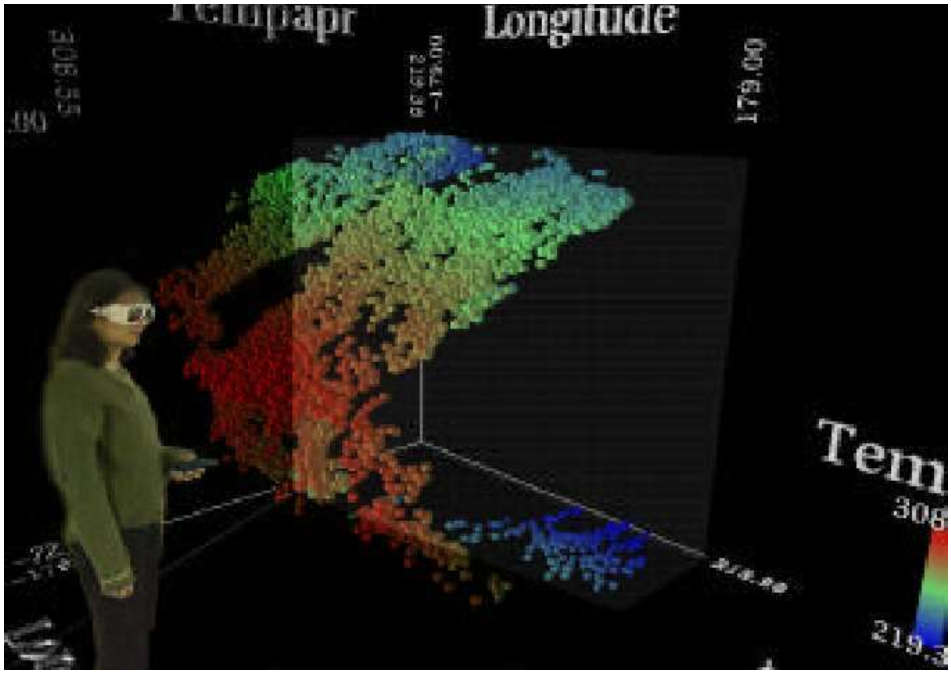


FIG.20 - Nuages de points 3D avec TIDE

La stéréoscopie à l'aide de lunettes commutées portées par l'utilisateur (FIG.20), est une technique classique en RV. Elle apporte une aide supplémentaire pour comprendre la structure 3D des données.

Dans l'outil *ARVis* développé lors de travaux de Thèse BLANCHARD (2005). Les objets graphiques représentent les règles d'association. Nous allons décrire cette métaphore de règles développée pour la phase de post traitement en ECD et la façon dont l'utilisateur peut interagir avec les données grâce à cet outil.

2.7.4 Exemple de visualisation de règles d'association : *ARVis*

Dans *ARVis*, le monde virtuel représente des sous-ensembles de règles, chaque règle est représentée par des figures géométriques, une sphère juchée sur un cône (FIG.22). Les mesures de qualités sont visualisées textuellement en plus d'être encodées graphiquement par des objets avec une taille, une luminosité et une position. La première version d'*ARVis* encode trois mesures de la façon suivante :

- la taille du cône pour la confiance,
- la taille de la sphère pour le support,
- la position de l'objet pour l'intensité d'implication.

La mesure d'intensité d'implication implémentée dans l'outil *ARVis* n'a pas encore été présentée. Cette mesure compare à partir d'une règle de la forme : $a \rightarrow b$, le nombre de contre-exemples $n_{a\bar{b}}$ observés dans les données au nombre de contre-exemples attendus sous l'hypothèse H_0 d'indépendance entre *a* et *b*. Le nombre de contre-exemples attendus sous H_0 est le cardinal de $X \cap \bar{Y}$ avec $|X| = n_a$ et $|Y| = n_b$. La règle est d'autant meilleure que la probabilité de produire plus de contre-exemples que les données est grande, BLANCHARD (2005).
Les objets sont positionnés dans une arène de telle sorte que les règles identifiées par des mesures de qualités faibles, soient placées dans le haut de l'arène. L'arène combine la hauteur avec la profondeur. Les objets les plus hauts dans l'arène seront donc également les plus éloignés. Ce mode de représentation apporte une première solution aux problèmes d'occultation.
La couleur des objets dans *ARVis* est une moyenne pondérée de la confiance et de l'intensité d'implication. La variable graphique de couleur apporte une évaluation synthétique de la qualité de la règle.
Un menu interactif permet d'utiliser huit relations de voisinage. La sélection d'une relation de voisinage sur une règle change le sous-ensemble actuel par un nouveau sous-ensemble qui contient toutes les règles voisines (FIG.21). Cette notion de voisinage doit faire sens pour l'utilisateur et peut être fondée sur une relation de similitude entre les règles ou bien sur la relation entre une règle et ses règles exceptions. Visuellement il change de monde, ce qui lui donne l'impression de naviguer dans l'ensemble de règles.

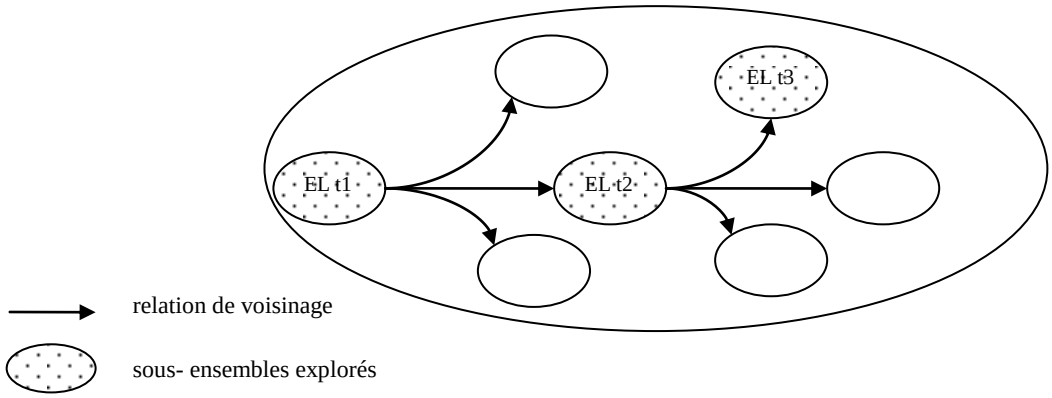


FIG.21 - La relation de voisinage pour naviguer parmi les sous-ensembles

A tout moment l'utilisateur peut revenir en arrière, c'est-à-dire au sous-ensemble précédent. Une relation de voisinages est donc un repère visuel qui déclenche les opérateurs de navigation dans les règles et les données. Comme le souligne CHEN (2004), ces repères visuels rendent la tâche de navigation plus intuitive. Ils facilitent l'acquisition des connaissances spatiales et la construction de la carte mentale de l'ensemble des données fouillées par l'utilisateur.

Pour chaque sous-ensemble extrait à l'aide des règles de voisinage, l'utilisateur est positionné dans le monde 3D devant la scène. Des primitives de navigation standards (marcher, voler, etc.) lui permettent alors de s'y déplacer librement pour explorer les règles et observer les détails de ces dernières (FIG.22 et 23).

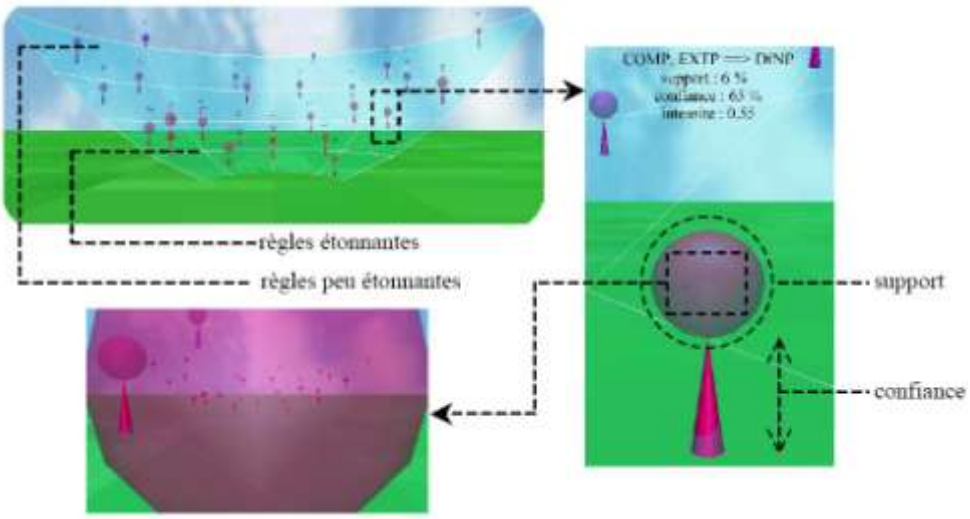


FIG.22 - Encodage graphique dans *ARVis*

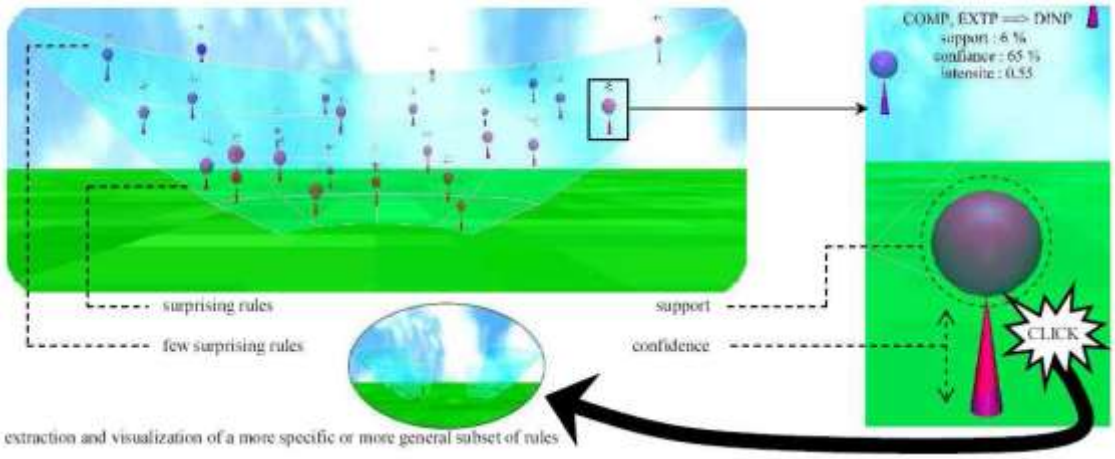


FIG.23 - Changement de monde dans *ARVis*

Chapitre 3

Etude préalable

3.1 Cahier des charges

3.1.1 Besoins

Cette étude traite conjointement deux parties :

- la production de règles d'association à partir d'un Système de Gestion de Base de Données (SGBD),

- la visualisation des règles extraites dans un environnement virtuel 3D.

La problématique principale est de créer une métaphore de règles représentant graphiquement les mesures de qualité suivantes :

- le support,
- la confiance,
- le lift (corrélation entre les *items* de la prémisse),
- le gain informationnel par attribut FREITAS A. A., (1998).

Le serveur de production de règles nécessite de mettre en œuvre des fonctions de contrôle et de statut de connexion ou de commandes et d'exécution de requêtes SQL. L'interface utilisateur doit comporter un dispositif de sélection d'*items* pour construire les requêtes. L'extraction est donc supervisée par l'utilisateur qui élabore une hypothèse de prémisse. Il indique le nombre d'*items* (*k-itemset*) et les attributs qui la composent. Nous rechercherons les règles à conclusion simple privilégiées par l'expert, c'est-à-dire ne comportant qu'un seul *item* dans la partie droite.

3.1.2 Modèle de données

	oid	outlook text	temp text	water text	windy text	rain text
1	5051053	sunny	hot	0.2	yes	hight
2	5051054	sunny	hot	0.6	yes	low
3	5051055	sunny	hot	0.4	yes	low
4	5051056	cloudy	cold	0.12	no	hight
5	5051057	sunny	cold	0.24	yes	medium
6	5051058	cloudy	fresh	0.13	no	hight
7	5051059	cloudy	fresh	0.25	no	low
8	5051060	cloudy	hot	0.2	no	low
9	5051061	foggy	cold	0.45	yes	low
10	5051062	foggy	fresh	0.12	yes	low
11	5051063	cloudy	cold	0.25	yes	medium
12	5051064	cloudy	cold	0.4	no	hight
13	5051065	sunny	hot	0.36	no	hight
14	5051066	sunny	fresh	0.21	yes	low
15	5051067	cloudy	hot	0.2	no	hight
16	5051068	foggy	hot	0.8	yes	low
17	5051069	sunny	hot	0.25	yes	medium
18	5163965	sunny	fresh	0.45	yes	low
19	5163966	cloudy	fresh	0.55	no	medium
20	5163967	cloudy	fresh	0.23	no	medium
*						

FIG.24 - Table du modèle de données.

A partir du modèle (FIG.24), une règle d'association sera notée :

- [R] : (Outlook = *sunny*, Temp = *hot*, Windy = *yes*) → (Rain = *low*)

Les couples variables/attributs sont séparés par une virgule qui exprime une relation conjonctive. Une flèche symbolise le sens de l'implication de la prémisse vers la conclusion.

Nous formalisons cette syntaxe pour l'écriture d'une règle d'association et retenons cet exemple pour la suite de notre étude. Nous validerons les résultats de l'application développée par comparaison avec les résultats de logiciels existants en libre utilisation (Tanagra, WEKA, etc.). Les calculs effectués par l'algorithme d'extraction pour le gain informationnel par attribut seront détaillés.

3.1.3 Environnement

Le système de fouille visuelle de données est construit sur une architecture client/serveur trois tiers (FIG.25) :

1. le serveur de production de règles d'association basé sur le SGBD relationnel *PostgreSQL*,

2. la couche applicative, un programme en langage C/C++ qui assure l'interface entre le système et l'utilisateur. L'application extrait les règles et écrit les résultats dans une table de la base de données. Les données d'entrée sont contenues dans un fichier codé au format CSV. Les règles extraites peuvent également être conservées sous le même format.

3. le client de visualisation lit la table de résultats et réalise l'encodage graphique des données dans une scène 3D.

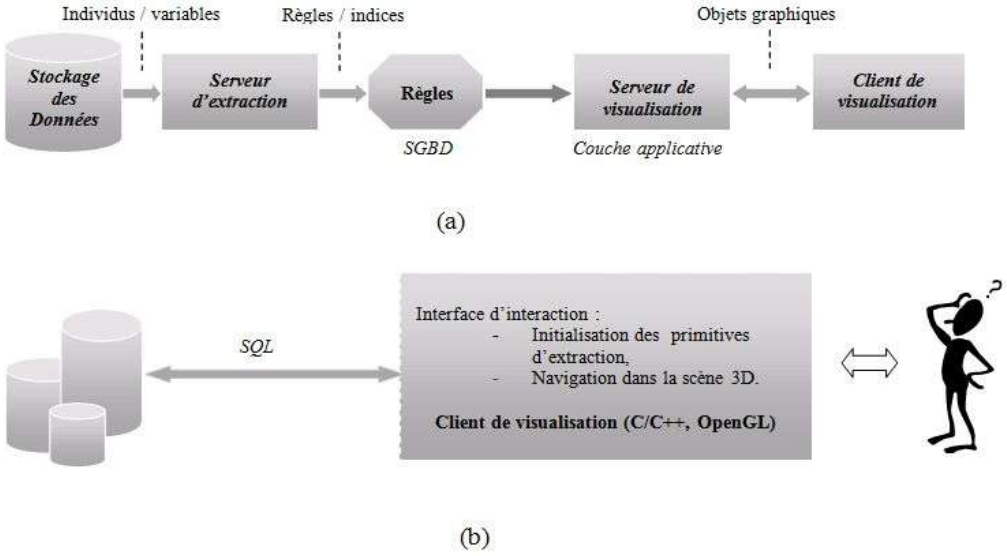


FIG.25 - Architectures logique (a) et physique (b)

3.1.4 Contraintes

La métaphore sera représentée en 3D à l'aide de la librairie graphique d'OpenGL. Cette librairie est totalement portable et offre de nombreuses fonctionnalités pour réaliser de la synthèse d'images. Mes encadrants ont décidé d'utiliser OpenGL afin de répondre entre autre, à la problématique de prise en charge des périphériques dédiés à la 3D (*drivers*). Nous présenterons et évaluerons de quelle façon OpenGL se distingue des autres technologies et notamment de Java3D.

Ergonomie :

1. pas de contraintes particulières pour l'interface de saisie, les interactions pour initialiser l'extraction des règles se feront en lignes de commande.

2. la possibilité de réaliser des essais successifs,

3. pour la visualisation : s'inspirer de la métaphore de la molécule.

Interopérabilité :

Utilisation de fichiers en entrée (données) et en sortie (règles) compatibles avec les normes et standards courants : PMML[27], fichiers CSV, etc.

Evolutivité :

1. autant que possible, utiliser un environnement de programmation standard,

2. code source suffisamment documenté et commenté pour faciliter l'intégration des évolutions ou améliorations futures.

3.2 Etude des solutions techniques

3.2.1 Choix du Système de Gestion de Base de Données (SGBD)

Une base de données doit offrir un vaste panel de fonctionnalités : déclencheurs, fonctions scalaires, etc. La richesse fonctionnelle des différents produits proposés par les éditeurs est variable. Bien que l'ensemble des fonctionnalités soit rarement nécessaire. En disposer de manière native représente un avantage pour les éventuelles évolutions futures. De nombreux Systèmes de Gestion de Base de Données (SGBD) sont disponibles sur le marché.
Certains sont proposés par des éditeurs établis de longue date, d'autres sont le fruit du travail de communautés de développeurs ou de nouvelles sociétés. La première catégorie regroupe les produits tels qu'Oracle ou Microsoft SQL serveur. Dans le second groupe se classent les acteurs du monde de *l'Open Source*[28] où *MySQL* et *PostgreSQL* se taillent une belle place auprès des entreprises[29].
Pour le SGBD à mettre en œuvre, je me limiterai aux produits sus mentionnés et mon choix sera guidé par les critères suivants :

1. utilisation libre des logiciels,

2. performance, fiabilité et fonctionnalités,

3. la portabilité du code,

[27] PMML, acronyme pour *Predictive Model Markup Language* est un langage basé sur XML définissant une manière standard de décrire les modèles statistiques et de traitement de données au sein des applications décisionnelles.
[28] Un logiciel *Open Source* est défini par l'organisation *Open Source Initiative* comme étant un logiciel libre, la réutilisation du code source est alors autorisée tant techniquement que légalement.
[29] http://www.scribd.com/doc/26799397/Programmez-N126-Janvier-2010

4. la possibilité de manipuler des grands volumes de données,

5. la richesse de la documentation fournie,

6. l'existence d'outils d'administration.

MySQL[30] vs PostgreSQL[31]

Une comparaison non exhaustive en termes de fonctionnalités et de performance pour *MySQL* et *PostregSQL* a été réalisée à partir de leur site Web respectif (TAB.8). Le choix de la distribution *Open Source* à mettre en œuvre s'appuiera sur les critères retenus de cette comparaison.

- License des logiciels

Les différences entre ces produits commencent avec les principes même qui les gouvernent, c'est-à-dire s'ils sont ouverts ou propriétaires. Microsoft SQL serveur ou Oracle avec leurs moteurs de stockage propriétaires et fermés sont donc fondamentalement différents des moteurs de stockage ouverts et extensibles dont disposent *MySQL* ou *PostgreSQL*.

1. *MySQL* est distribué sous licence GPL (*General Publique License*), un principe qui n'interdit pas de faire payer l'accès à l'œuvre mais qui offre, une fois celle-ci obtenue, des garanties de liberté pour la modifier, l'étudier ou la redistribuer.

2. *PostgreSQL* est distribué sous licence BSD (*Berkeley software licence*), cette licence autorise la réutilisation de tout, ou partie du logiciel sans restriction, que celui-ci soit intégré dans un logiciel propriétaire ou libre. Les termes qui composent cette licence respectent en tous points les termes de la licence GPL. Cette observation n'étant pas complètement symétrique. La licence BSD fait donc partie des licences les moins restrictives du monde de l'informatique. Elle se rapproche de la notion de domaine publique.

- Fonctionnalités et performances

Il est essentiel de considérer les besoins en termes de performance, de fiabilité, de sécurité, etc. pour toutes les applications qui s'appuient sur une base de données. Pour notre serveur de production de règles, le critère de sélection principal étant de pourvoir contenir de grands volumes de données.

[30] http://www-fr.mysql.com/
[31] http://www.postgresql.org/

Fonctionnalités	*PostgreSQL*	*MySQL*
ANSI SQL conformité	Très près du standard ANSI SQL	Ne suit que quelques standards de l'ANSI SQL
Performance	Lent	Rapide
Sous-requêtes	Oui	Non
Transactions	Oui	Oui, mais avec utilisation d'une table d'InnoDB
Réplication de base de données	Oui	Oui
Support clés étrangères	Oui	Non
Vues	Oui	Non
Procédures stockées	Oui	Non
Triggers	Oui	Incomplet
Unions	Oui	Non
Full joins	Oui	Non
Contraintes	Oui	Non
Windows support	Oui	Oui
Vacuum (clean up)	Oui	Non
ODBC	Oui	Oui
JDBC	Oui	Oui
Différents types de tables	Non	Oui

TAB.8 - Fonctionnalités *PostgreSQL* et *MySQL*

1. *MySQL* a pour vocation d'être une solution plus facile, rapide et extrêmement optimisée qui *a contrario* de *PostgreSQL* n'implémente pas toute la puissance du relationnel. *MySQL* a su s'imposer avec son fameux modèle LAMP (*Linux*, *Apache*, *MySQL*, *PhP*) pour supporter les sites Web dynamiques et vise la performance comme critère essentiel (TONIC F., (2010).

2. *PostgreSQL* est un système de base de données relationnelle de référence complet, qui suit le standard ANSI[32]. Ce système offre de nombreuses fonctionnalités génériques pour des applications de base de données traditionnelles. *PostgreSQL* par rapport à *MySQL* est la base de données la plus conforme au standard SQL.

[32] Acronyme pour *American National Standards Institute* (ANSI) est un organisme privé qui supervise le développement de normes pour les produits, les services, les procédés, les systèmes et les employés des États-Unis

PostgreSQL possède les capacités de gérer de très gros volumes de données et repose sur un modèle orienté objet, une technologie qui ouvre des horizons beaucoup plus larges qu'un dispositif classique tel que celui de *MySQL*. Bien que *MySQL* soit plus facile et rapide à mettre en œuvre, nous retiendrons *PostgreSQL* pour monter notre serveur de production de règles.

- *PostgreSQL* : portabilité

PostgreSQL est disponible pour de nombreuses plates-formes (Windows, Linux, Mac OS, etc.). La librairie *libpq* est l'interface de programmation d'application C, de *PostgreSQL*. Le ficher d'en-tête *lippq-fe.h*, peut être compilé pour le développement des applications clientes et porté d'un système d'exploitation à l'autre.

- *PostgreSQL* : documentation et outils d'administration

Les fonctions[33] disponibles pour *PostgreSQL* sont clairement documentées. Le code fourni est largement commenté. L'intégration du code dans notre programme client se fera donc sans difficultés apparentes. Les échanges de requêtes avec le serveur se feront en mode natif. La programmation au niveau natif assure d'après ZAHER (2002), la possibilité d'exploiter toutes les spécificités de la base de données.
L'outil d'administration *pgAdmin*[34] (FIG.26) peut se connecter à toutes bases de données PostgreSQL 7.3/7.4 et 8.x en utilisant la bibliothèque native *libpq*. L'application n'a donc pas besoin d'une couche *ODBC*[35]ou *JDBC*[36]supplémentaire. Cet outil comporte de nombreuses fonctionnalités (éditeur de requêtes SQL, éditeur de tables, scripts de requêtes, etc.). J'utiliserai donc la version 8.4 de *PostgreSQL* et son outil d'administration *pgAdmin* III v1.10.1.

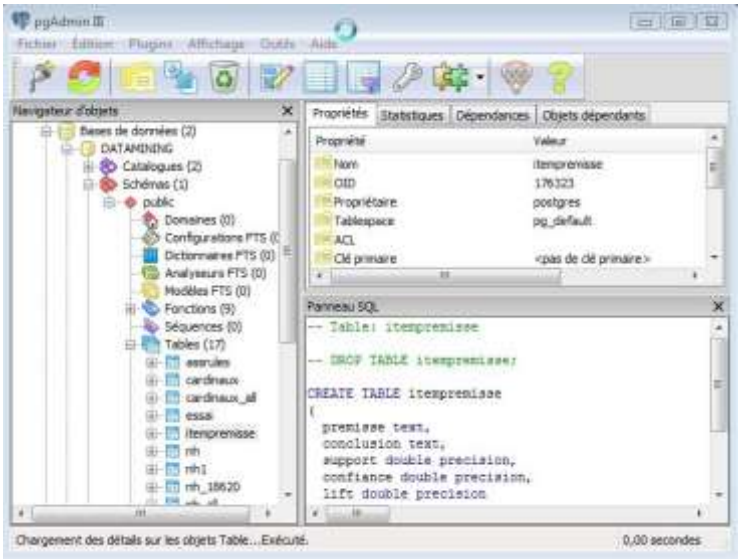


FIG.26 - *PgAdminIII* : outil d'administration pour *PostregSQL*

[33] http://www.postgresql.org/docs/manuals/
[34] *www.pgadmin.org/*
[35] Acronyme pour *Open DataBase Connectivity*, c'est un ensemble API/pilote défini par Microsoft permettant la communication entre des clients de bases de données fonctionnant sous Windows et les systèmes de gestion de base de données du marché.
[36] Acronyme pour *Java DataBase Connectivity*, c'est une API développée pour les programmes utilisant la plateforme Java.

3.2.2 Extraction des règles

Étant donné un ensemble T de transactions de cardinal $n = |T|$ et soit une règle de la forme :

$X \rightarrow y$, où X est un *itemset* $X \subseteq T$ et y est un *item* $\in T \setminus X$ (conclusion simple). La recherche de règles d'association consiste alors à trouver tous les *items* y tels que :

$Y/y \in \{T\}$ et $y \notin X$; $y \neq x$.

$|X| = n_x$,

$|Y| = n_y$,

$|X \cup Y| = n_{x,y}$.

- Une primitive d'extraction se décompose en deux étapes :

    3. l'extraction de tous les *itemsets* qui interviennent dans les règles : relations combinatoires entre les attributs par variables.

    4. la construction des règles qui consiste à donner leur syntaxe et la cardinalité des attributs qui les composent (n_x, n_y, $n_{x,y}$).

Pour notre étude, l'utilisateur spécifie lui-même les *items* et les variables. Les règles spécifiées possèdent donc toutes le même *itemset* de prémisse et les règles générales seront toutes construites à partir d'un seul *item*.

3.2.3 Technologie 3D

- OpenGL : un standard de fait

La spécification OpenGL est une interface de programmation standardisée de référence pour le rendu 3D, elle a été développée par un consortium d'industriels ayant des intérêts dans le domaine (Intel, AMD, Apple, etc.). La bibliothèque OpenGL est basée sur le langage C. La technologie OpenGL offre également l'avantage comme *PostgreSQL*, d'être portable sur de nombreuses plateformes.

- OpenGL : principe de base

OpenGL ne se charge pas directement de l'affichage et décrit seulement des objets tridimensionnels, le rendu OpenGL s'initialise soit :

1. par l'API[37] du système d'exploitation,

2. par l'API spécifique GLUT (*OpenGL Utility Toolkit*).

[37] API acronyme pour *Application Programmable Interface*, traduit par interface de programmation.

GLUT est une bibliothèque co-existante avec OpenGL. Elle contient les routines de bas niveau pour gérer les matrices de transformation et de projection, la facettisation des polygones et le rendu de surface. C'est une boite à outils indépendante du système de fenêtrage. GLUT est une bibliothèque qui simplifie plus encore la tâche de portage entre les différents systèmes d'exploitation.

La fenêtre graphique est définie de manière unique dans un contexte d'affichage (*device context*). Une procédure appelée *handle* permet au programme d'accéder à la fenêtre et d'y passer les ordres de dessin directement au pipe-line de rendu (*rendering context)*. OpenGL se contente alors de « mixer » les informations et de les transmettre au système d'exploitation. Cela explique la raison pour laquelle, la librairie graphique OpenGL est 100% portable.

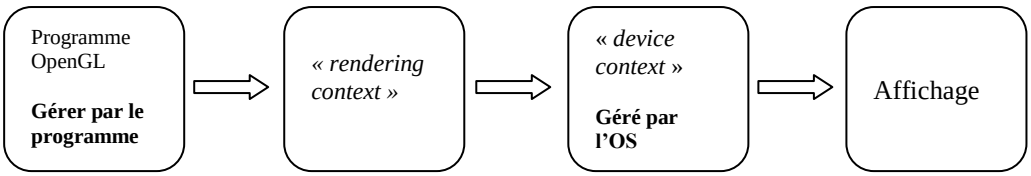


FIG.27 - Principe d'affichage OpenGL

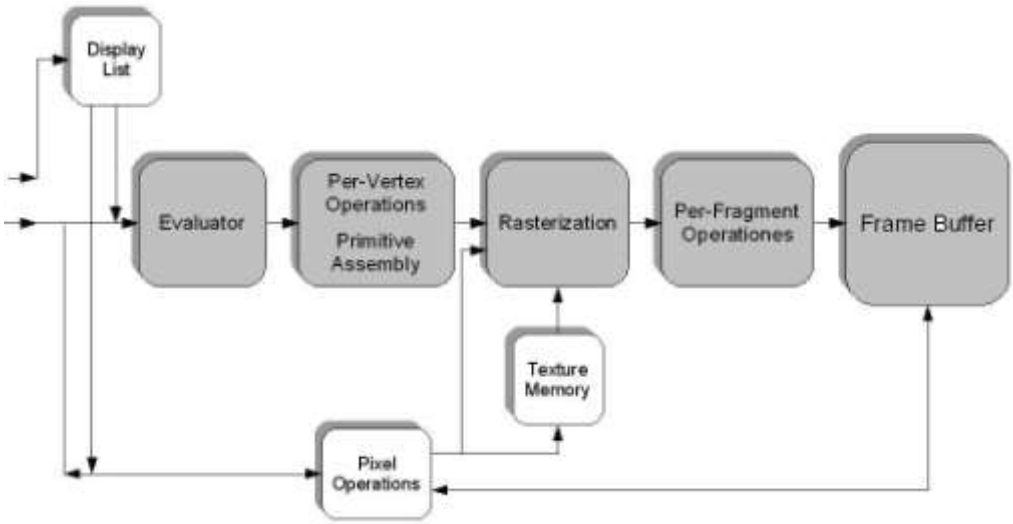


FIG.28 - Pipe-line de rendu simplifié OpenGL

1. les évaluateurs produisent une description des objets à l'aide de sommets et de facettes. Les opérations sur les sommets sont les transformations spatiales (rotations et translations).

2. l'assemblage de primitives regroupe les opérations de *clipping* qui consiste à éliminer les primitives en dehors d'un certain espace de transformation perspective.

3. La discrétisation (*rasterization*) est la transformation des primitives géométriques en fragments correspondant aux pixels de l'image. Les opérations sur les fragments vont calculer chaque pixel de l'image en combinant les fragments qui se trouvent à l'emplacement du pixel. On trouve par exemple la gestion de la transparence et le *Z-buffer* (pour l'élimination des surfaces cachées).

4. le Frame Buffer est une zone de la mémoire vidéo dans laquelle l'image est écrite (sous forme d'un bitmap) avant d'être envoyée vers le moniteur

La contrainte pour le système d'exploitation est alors de pouvoir dialoguer avec OpenGL. Ces dialogues sont nécessaires pour assurer les interactions de l'utilisateur avec les objets de la scène 3D. Cette liaison est réalisée par exemple pour MS Windows, grâce au driver *Opengl32.dll*[38].

- Java3D :

Java3D est une librairie de visualisation en trois dimensions développée par Sun[39]. Java 3D offre tous les outils nécessaires pour la génération d'objets complexes, le rendu, l'éclairage, et la navigation dans l'univers créé. Une scène Java3D est conçue comme une famille arborescente d'objets graphiques. Il s'agit d'un graphe acyclique qui chaine les objets en assurant la gestion des déplacements dans la scène et le contrôle du point de vue.

L'arborescence se construit à partir d'un nœud racine (*La locale*) auquel on ajoute des nœuds enfants par des méthodes de type *add* (*BranchGroup*). Bâtir une scène 3D commence par l'instanciation des éléments (objets 3D) qui constituerons la scène. Ces objets sont ensuite regroupés et peuvent être modifiés (apparence, couleur, taille, etc.). Enfin, les différentes scènes construites de façon séparée seront placées dans un conteneur commun nommé l'univers virtuel. Java3D se charge du rendu à l'écran.

On trouve généralement un seul univers virtuel collectionnant toutes les scènes 3D (FIG.29). La « *locale* » est le nom donné au système de coordonnées orthonormées directes (main droite) dans le monde virtuel, c'est un repère pour poser une ou plusieurs scènes3D (*BranchGroup*).

Un *Shape* 3D est un *LeafNode* qui possède une géométrie (forme) et une apparence (couleur, transparence, etc.). Le « *TransformGroup* » défini un nouveau système de coordonnées pour les objets dans la hiérarchie. Il permet par rapport au nœud parent de l'objet, d'effectuer une translation, une rotation ou une homothétie.

[38] DLL acronyme pour *Dynamic Link Library* traduit par bibliothèque de liens dynamiques.
[39] http://www.javasoft.com/products/javamedia/3D/index.html

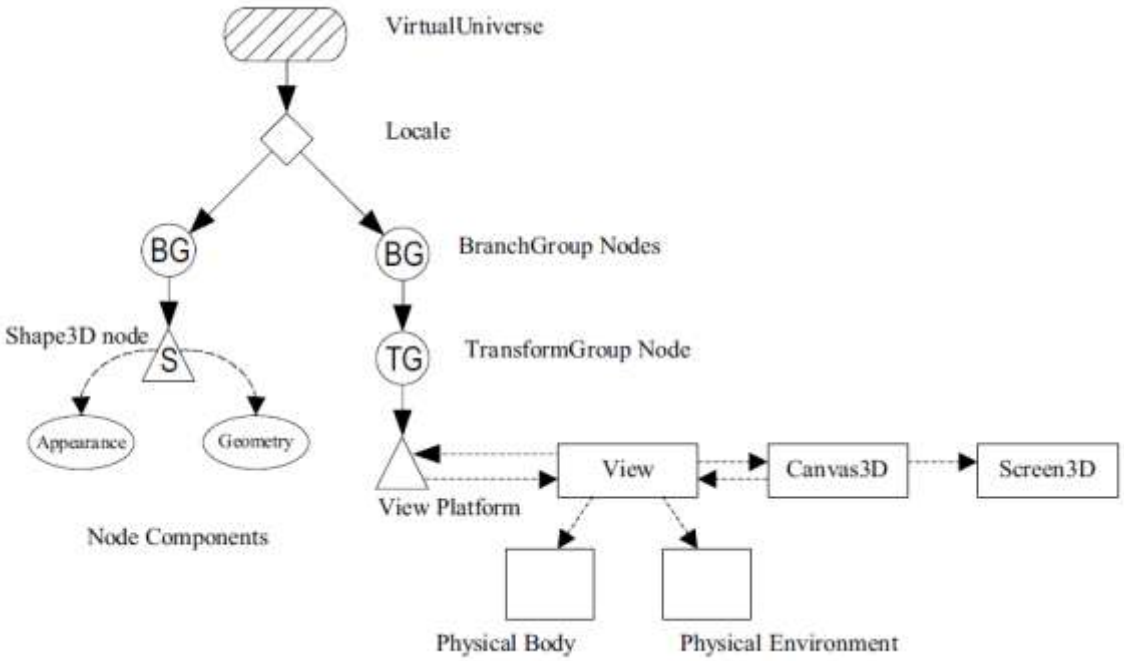


FIG.29 - Arborescence des objets d'un univers Java3D

Il n'existe peu d'ouvrages pour Java3D et bien que quelques sites spécialisés en infographie s'intéressent également à cette technologie, la meilleure source actuelle reste encore le site de Sun.

OpenGL vs Java3D

Certaines fonctionnalités de l'API pour utiliser Java3D font appel à des méthodes natives de la *DLL*, *Java3D/OpenGL* [GL4J]. L'objectif de cette couche logicielle est d'utiliser une bibliothèque dans un langage de programmation différent de celui avec lequel elle a été écrite. Cela revient donc à utiliser une bibliothèque native enveloppante qui traduit la spécification du langage de développement utilisé dans le langage supportant la bibliothèque graphique OpenGL. L'efficacité d'une telle méthode dépend alors des capacités de cette bibliothèque à lier les deux langages.

Programme Java de visualisation 3D
API Java proposant des routines 3D
Bibliothèques 3D de base (OpenGL, Direct 3D[40])
Hardware : CPU + entrées (périphériques 3D) + sorties écran carte graphique avancée

TAB.9 - Architecture logicielle, Java3D

[40] Direct 3D est un composant propriétaire de l'API Microsoft DirectX contrairement à OpenGL qui est une spécification libre et gratuite.

OpenGL offre très peu de commandes haut niveau pour décrire des objets 3D. Son utilisation nécessite de construire les modèles graphiques à partir d'un jeu restreint de formes primitives (points, lignes, polygones, etc.). Java 3D intègre directement ces modèles d'objets dans des méthodes. La mise en œuvre est rapide et la prise en main ne nécessite pas de posséder de larges connaissances en programmation 3D. Nous avons vu qu'il était également nécessaire d'utiliser une couche logicielle supplémentaire.

Toutes les fonctionnalités d'OpenGL ne sont malheureusement pas totalement supportées par cette couche appelée *binding* Java-OpenGL. Par exemple *JAVAGL*, l'un des premiers paquetages et maintenant abandonné supportait seulement 20% des fonctions OpenGL. D'après ABISROR (2002), le paquetage *JOGL*, probablement l'un des plus avancé supporte 60 à 70% des fonctions d'OpenGL mais n'est plus mis à jour depuis décembre 1997.

Les avantages d'OpenGL face à Java3D sont :

1. de fournir un modèle procédural pour le graphisme qui correspond à beaucoup d'algorithmes et de méthodes que les programmeurs de graphiques ont toujours utilisées,

2. un accès direct au pipeline de rendu et cela est vrai pour n'importe quel *binding* de langage,

3. que les distributeurs de chaque type de matériel 3D supportent OpenGL,

4. qu'OpenGL est un standard de fait avec lequel les distributeurs évaluent leur technologie graphique,

5. que la spécifications OpenGL est basée sur le langage C et est normalisée.

Java3D et la technologie VRML[41] utilisée dans *ARVis* possèdent des commandes « haut niveau » pour la création d'objets 3D. Ce type de commandes n'existe pas pour OpenGL mais ne sera pas nécessaires pour notre métaphore 3D (molécule). Cette absence ne sera donc pas délétère pour le projet.

La conception d'OpenGL comme une interface rationnelle indépendante du matériel devrait permettre une implémentation plus facile des *drivers* pour les périphériques dédiés à la 3D.
Ces périphériques sont nécessaires pour assurer l'interaction de l'utilisateur avec les objets représentés dans la scène. L'interaction pour un utilisateur plongé en milieu immersif est un facteur essentiel de réalité virtuelle.

[41] VRML acronyme pour *Virtual Reality Markup Language*, c'est un langage interprété de description d'univers virtuels en trois dimensions.

3.2.4 Langage de programmation et environnement de développement

- Langage

Le prototype de fouille de donnée est développé en C/C++. Les librairies graphiques OpenGL seront intégrées dans le code du programme. Le code est portable sur différentes plateformes matérielles. Le développement de cette application au niveau natif nous assure d'obtenir des performances nominales, quelque que soit le système d'exploitation utilisé. Cependant, le C reste un langage complexe et la phase de débogage peut être longue.

- Outils de travail

Après l'élimination des solutions propriétaires nous utiliserons l'environnement de développement: Code::Blocks. Un logiciel écrit en C et libre (gratuit). Cet environnement est conçu pour être extensible (*plugins*) et entièrement configurable. De nombreuses librairies sont préinstallées pour faciliter l'initialisation de projets comme OpenGL. L'interface proposée est identique à celle de MS-Visual Studio. Bien que Code::Blocks fût à l'origine développé pour Linux, le code source des programmes réalisé peut être compilé pour MS-Windows. Code::Blocks intègre à cet effet un compilateur minimaliste (*mingw*[42]) dont les sources complètes sont fournies avec un ensemble d'outils appropriés pour le développement d'applications *win32*.

3.2.5 Récapitulatif des choix techniques

Les choix techniques retenus sont récapitulés dans le tableau suivant :

Fonctionnalités	Outils
Serveur d'extraction de règles	*PostgreSQL v8.4*
Outil d'administration serveur	*PgAdminIII*
Langage de programmation	*C/C++*
Compilateur	*wingw*
Environnement de développement	*Code::Blocks*
Visualisation en 3D	*OpenGL*
Fichiers d'échange	*Fichiers CSV*

TAB.10 - Récapitulatif des choix techniques

[42] http://www.mingw.org/ /

3.3 Proposition d'une nouvelle métaphore 3D

3.3.1 Présentation de la métaphore

D'un commun accord avec les encadrants, nous avons décidé d'utiliser la métaphore de la molécule pour représenter les règles d'association (FIG.30). Chaque règle est symbolisée par un graphe composé de sphères qui matérialisent les *items*. Nous nous intéresserons aux règles à conclusion simple, c'est à dire ne comportant qu'un seul *item*. Les liaisons entre les sphères se croisent en un point défini par le centre des coordonnées (x, y, z) des différentes sphères. Le centre du graphe de prémisse sera relié à l'*item* de conclusion par un seul arc.

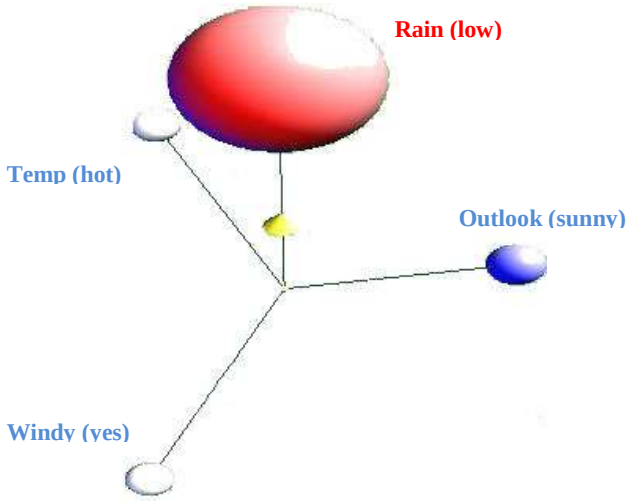


FIG.30 - Métaphore d'une règle d'association

La métaphore que nous proposons implémente les concepts suivants :

1. représentation de l'implication brute de chaque *item* : le diamètre des sphères varie en fonction d'une mesure basée sur la théorie de l'information.

2. relations inter-*items* : la relation doit indiquer si la présence d'un *item* favorise la présence d'un autre item au sein de son ensemble (prémisse). La relation existante entre deux *items* sera représentée par la distance qui sépare deux *items*. Cette distance sera évaluée en se basant sur la mesure du lift.

3. représentation des mesures d'intérêt support et confiance :

    - l'indice de support pour positionner les règles dans une scène 3D. La scène est une arène inspirée de l'outil *ARVis*.

    - l'indice de confiance pour traduire la distance qui sépare la prémisse de la conclusion.

3.3.2 Calcul des indices de qualité

- Indices de support et confiance

Les indices de support et de confiance ont déjà été définis au §2.2.8 :

6. *Support* $(X \rightarrow Y) = \frac{P(XY)}{n} = \frac{|X \cup Y|}{|T|} = \frac{nx,y}{n}$

7. *Confiance* $(X \rightarrow Y) = \frac{P(XY)}{P(X)} = \frac{|X \cup Y|}{|X|} = \frac{nx,y}{nx}$

- Calcul du lift

Le lift a déjà été défini au §2.2.8.

$$Lift\ (X \rightarrow Y) = \frac{P(XY)}{P(X)\,P(Y)} = \frac{Confiance\ (X \rightarrow Y)}{Support\ (Y)} = \frac{|X \cup Y|}{|T|} * \frac{|T|}{|Y|}$$

Le lift désigne les liens de corrélation existants entre les différents *items* de la prémisse. Un lift élevé indique une corrélation forte entre deux *items*. Le lift n'est pas une mesure probabiliste. Il sera donc nécessaire d'homogénéiser sa valeur vis-à-vis des autres indices pour la représenter graphiquement. Nous utiliserons alors l'inverse du lift variant sur un intervalle [0,1]. Une corrélation forte entre deux *items* sera représentée par une distance faible qui les sépare. En prenant l'inverse du lift, la distance qui sépare deux *items* sera modélisable pour toutes les règles. Nous l'appelons *InvLift*.

Distance séparant deux *items* : *InvLift* $(X \rightarrow Y) = \frac{1}{Lift\ (X \rightarrow Y)}$

- Calcul du gain informationnel par attribut, FREITAS A.A., (1998)

Le gain informationnel par attribut montre l'importance de considérer individuellement chaque attribut qui compose une règle d'association. Pour cela, il donne une évaluation de l'intérêt que l'on peut porter à chacun d'eux. Cette mesure est peu répandue dans la littérature mais présente néanmoins un concept clé en fouille de données pour évaluer l'interaction des attributs entre eux. Nous chercherons par l'implémentation du gain informationnel à augmenter le niveau de granularité dans l'exploration des données par rapport aux autres outils de visualisation existants de règles d'association. Nous présentons cette mesure.

Pour une règle $A \rightarrow G$, le calcul du gain informationnel *InfoGain* (A_i), pour chaque attribut A_i de la prémisse s'effectue en utilisant les formules (1), (2) et (3).

Dans ces formules :

1. Info(G) est l'information fournie par les attributs de la classe de G, c'est-à-dire par les attributs de la conclusion,
2. ($G|A_i$) est l'information fournie par les attributs de la classe de G par rapport aux attributs A_i (prémisse),
3. A_{ij} désigne la valeur du j-ème attribut A_i (prémisse),
4. G_j désigne la valeur du j-ème attribut G (conclusion),
5. Pr(A) est la probabilité de (A) et Pr(A|G) désigne la probabilité conditionnelle de A sachant G.

L'indice « j » dans les formules (2) et (3) varie sur l'intervalle [*1*.. *n*], où « n » représente les cardinalités des valeurs potentielles pour la variable respective de l'*item* en conclusion. L'indice *k* dans la formule (3) varie sur l'intervalle [*1*.. *m*], où « m » est le nombre de valeurs de l'attribut A_i dans la prémisse.
Dans (2) nous faisons une mesure d'entropie sur les attributs de la conclusion. Dans (3) cette mesure d'entropie est conditionnée par les valeurs de A_i.

$$\text{InfoGain}(A_i) = \text{Info}(G) - \text{Info}(G|A_i) \quad (1)$$

où

$$\text{Info}(G) = -\sum_{j=1}^{n} \Pr(G_j) \log \Pr(G_j) \quad (2)$$

$$\text{Info}(G|A_i) = \sum_{k=1}^{m} \Pr(A_{ik}) \left(-\sum_{j=1}^{n} \Pr(G_j|A_{ik}) \log \Pr(G_j|A_{ik}) \right) \quad (3)$$

3.3.3 Exemple explicatif

A partir du modèle de données présenté §3.1.2, nous allons mettre en application les différentes mesures de qualité pour notre exemple de règle, que l'on rappelle :

[R]: (Outlook = *sunny*, Temp = *hot*, Windy = *yes*) → (Rain = *low*)

Nous renommons [R] pour simplifier la syntaxe : *R* : (O = *s*, T = *h*, W = *y*) → (R = *l*)

Pour cette règle, on trouve également (TAB.11) les cardinalités des *items* et *itemsets*. Nous combinons deux à deux les différents *items* de la prémisse entre eux (FIG.31). Nous évaluons ensuite la mesure du lift par les associations d'*items* réalisées. Pour cela, nous trouvons également les cardinalités de chacun des attributs qui interviennent dans ces règles plus générales.

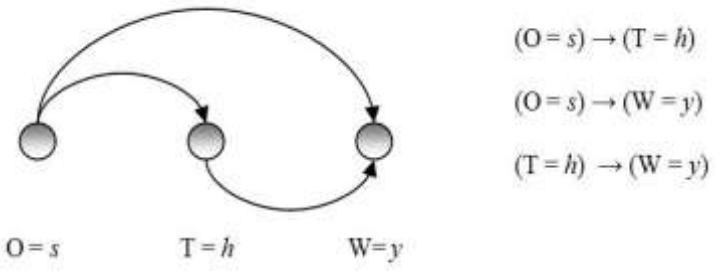


FIG.31 – Combinaison des items de la prémisse

r(*X*)	n	n_x	n_y	$n_{x,y}$
(O = *s*, T = *h*, W= *y*) → (R=*l*)	20	4	9	2
(O = *s*) → (T = *h*)	20	8	8	5
(O = *s*) → (W = *y*)	20	8	11	7
(T = *h*) → (W = *y*)	20	8	11	5

TAB.11 - Cardinalités des attributs

Indices	détails	résultats
$Support_{(X) \to (Y)}$	$\frac{2}{20} = 0.1$	10%
Confiance $_{(X) \to (Y)}$	$\frac{2}{4} = 0.5$	50%
Lift $_{(X) \to (Y)}$	$\frac{2}{20} / (\frac{4}{20} * \frac{9}{20}) = \frac{0.1}{0.09}$	1.11

TAB.12 - Résultats des indices de qualité ($X \to Y$)

Lift $_{(O = s) \to (T = h)}$	$\frac{5}{20} / (\frac{8}{20} * \frac{8}{20})$	1.56
Lift $_{(O = s) \to (W = y)}$	$\frac{7}{20} / (\frac{8}{20} * \frac{11}{20})$	1.59
Lift $_{(T = h) \to (W = y)}$	$\frac{5}{20} / (\frac{8}{20} * \frac{11}{20})$	1.14

TAB.13 - Calcul du lift entre les attributs

Nous détaillons la recherche du gain informationnel avec pour exemple le premier attribut de la prémisse contenu dans la règle *R1 :* la variable « Outlook » associée à l'attribut « *sunny* ». Le mode de calcul sera identique pour tous les autres attributs contenus dans la prémisse de la règle *R*, (Temp = *hot*, Windy = *yes*).
Nous adoptons une méthode binaire pour calculer le gain informationnel de chacun des attributs de la règle. C'est-à-dire que pour la règle *R*, nous considérons la variable « Rain » de la conclusion avec les valeurs *low* et *no_low* (*l ; no_l*) comme attributs. Pour la variable « Outlook » retenu en exemple, nous prendrons les valeurs *sunny* et *no_sunny* (*s* ; *no_s*).

InfoGain (O = s) = InfoGain (R = *l*) - InfoGain (R = *l* | O = *s*)

(1) InfoGain (R = *l*) = - [*Pr* (R = *l*) * log (*Pr* (R = *l*))

+ *Pr* (R = *no _l*) * log (*Pr* (R =*no _l*))]

(2) InfoGain (R = *l* | O = *s*) = *Pr* (O = *s*) * [- (*Pr* (R= *l* | O = *s*) * log (*Pr* (R= *l* | O = *s*))

+ *Pr* (R= *no_l* | O = *s*) * log (*Pr* (R= *no_l* | O = *s*))]

+ *Pr* (O = *no_s*) * [- (*Pr* (R = *l* | O = *no_s*) * log (*Pr* (R = *l* | O = *no_s*))

+ *Pr* (R= *no_l* | O = *no_s*) * log (*Pr* (R= *no_l* | O = *no_s*))]

- Application numérique :

cardinalités / attributs/variables	(A_i)	(G_j)	(G_j | A_i)
(R = *l*)	/	9	/
(R = *no_l*)	/	11	/
(O = *s*)	8	/	/
(O = *no_s*)	12	/	/
(R = *l* | O = *s*)	8	9	**4**
(R = *l* | O = *no_s*)	12	9	**5**
(R = *no_l* | O = s)	8	11	**4**
(R= *no_l* | O = *no_s*)	12	11	**7**

TAB.14 - Cardinalités des ensembles

(1) Info (Rain = *l*) = - [($\frac{9}{20} * \log\ (\frac{9}{20})$) + ($\frac{11}{20} * \log (\frac{11}{20})$)] = 0.29

(2) Info (Rain = *l* | O = *s*) = $\frac{8}{20}$ * [- (($\frac{4}{20} * \log (\frac{4}{20})$) + ($\frac{4}{20} * \log (\frac{4}{20})$))]

$$+ \frac{12}{20} * [- ((\frac{5}{20} * \log (\frac{5}{20})) + (\frac{7}{20} * \log (\frac{7}{20})))] = 0.18$$

3.3.4 Encodage graphique des mesures d'intérêt

- Le gain informationnel

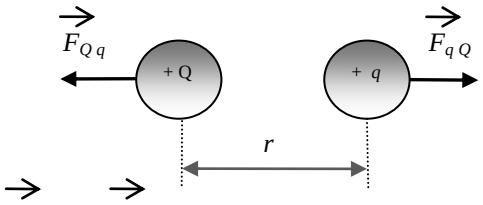
Le gain informationnel de chacun des attributs représentera le diamètre des sphères dans l'espace 3D. Nous observons que pour les mêmes *itemsets*, le gain informationnel de chacun des attributs varie de façon significative en fonction de l'*item* de la conclusion. Etant donné la nature statistique de cette mesure et dans le cas où nous serions en présence de variables catégoriques et continues. Le gain informationnel qu'affecte cette variable pourrait alors varier sur l'intervalle]- ∞, +∞ [.

Il sera donc nécessaire d'effectuer une normalisation de cette mesure par rapport à l'échelle de la scène 3D. Nous limiterons ainsi la représentation des sphères dont les diamètres trop élevés surchargeraient la représentation ou trop faible qui seraient à peine perceptibles pour l'utilisateur. Lorsqu'un attribut de la prémisse présente un gain informationnel négatif, les sphères sont de couleur blanche.

- Le lift

Comme nous l'avons déjà spécifié, les corrélations existantes entre les *items* seront représentées par les distances qui les séparent. Nous utiliserons deux lois de la physique fondamentale pour élaborer ces distances :

1. la loi de Coulomb : elle exprime la force de l'interaction entre deux particules chargées électriquement.



$$|F_{Q\,q}| = |F_{q\,Q}| = k\,\frac{|q\,.\,Q|}{r2}$$

FIG.32 - Loi de Coulomb, répulsion[43]

[43] http://en.wikipedia.org/wiki/Coulomb's_law

Pour toutes les particules (sphères) de la métaphore 3D. Les charges électriques seront de mêmes signes, donc chargées positivement ou négativement. Le but recherché étant de créer une force de répulsion appliquée à une charge q_1 par la présence d'une autre charge q_2. Cette force est proportionnelle au produit des deux charges et inversement proportionnelle au carré de la distance qui les séparent. La force de répulsion de Coulomb est donnée par la formule suivante :

$F = k_c \frac{q1 \,.\, q2}{r2}$; avec $k_c = \frac{1}{4\,\pi\,\epsilon 0} = 8.86\ 10^{-9}$ Constante de Coulomb.

La seule force de répulsion aurait pour effet d'éloigner indéfiniment les sphères les une par rapport aux autres. Il est alors nécessaire de créer une force qui s'y oppose.

2. la loi de Hooke : c'est une loi de comportement des solides lorsque ceux ci sont soumis à une déformation élastique de faible amplitude. Elle indique que la force appliquée à un solide pour le déformer est proportionnelle à l'extension qu'il subit. Hooke met en avant la théorie des ressorts. Deux aspects différents sont alors importants lorsque ces ressorts sont soumis à une force croissante :

   - la linéarité qui exprime que l'allongement d'un ressort est proportionnel à la force qu'il subit.

   - l'élasticité qui exprime que cet effet est réversible, c'est-à-dire que si la force disparait, le ressort revient à sa position d'origine. Notons que l'élasticité admet cependant une limite qui est indépendante de la notion de linéarité. Hooke considère seulement la phase élastique et linéaire, donc proportionnelle et réversible.

Pour le ressort (FIG.33), on donne son allongement L par (l - l_0) ou l_0 correspond à la longueur du ressort à vide et l, à la longueur du ressort étiré ou comprimé lorsque celui-ci est soumis à une force F. Cette force s'exprime par la formule : $F = k\ (l - l_0)$ avec k : constante de raideur.

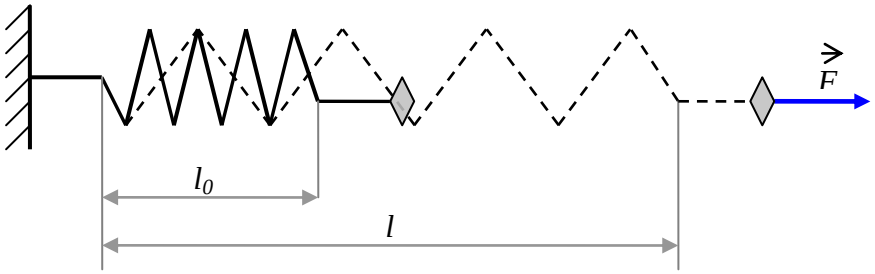


FIG.33 - Effet de Hooke, attraction[44]

[44] http://en.wikipedia.org/wiki/Hooke's_law

L'objectif de la mise en œuvre de ces deux lois physique est de définir un système d'attraction-répulsion entre les sphères (FIG.34). Nous représentons ici, la force appliquée à la première sphère (q_1) et procéderons comme suit :

1. Dans un premier temps nous attribuons des poids aux différents sommets du graphe. Ces poids représentent la charge électrique de chacune des sphères. Puis nous calculons un vecteur de répulsion grâce à la formule de Coulomb pour disperser les sphères dans l'espace 3D.

2. Dans un deuxième temps, nous appliquerons à chaque sphère un vecteur d'attraction basé sur l'effet de Hooke. Cela revient pour une sphère, à positionner un ressort (un arc) entre celle-ci et toutes les autres.

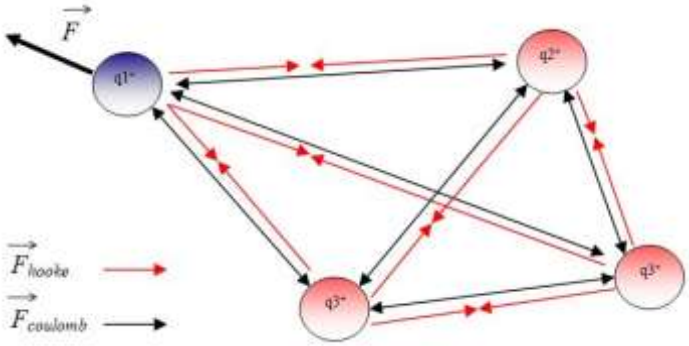


FIG.34 - Attraction, répulsion (q_1)

Dans ce montage, la position de chacune des sphères dépend alors de la somme des forces répulsives des autres sphères, et de la somme des forces attractives auxquelles elles sont associées. Nous réalisons un graphe complet. En théorie des graphes, un graphe complet est un graphe possédant n sommets tous reliés deux à deux par une arête. Le nombre d'arrêtes pour un graphe K_n est alors donné par :

$$\sum_{i=1}^{n}(n-i) = \frac{n(n-1)}{2}$$

Le premier terme s'obtient par la suppression d'un premier sommet de K_n qui entraine la suppression de $n-1$ arêtes, puis la suppression d'un deuxième sommet, la suppression de $n-2$ arêtes, et celle d'un i-ème sommet de $n-i$ arêtes.

L'algorithme de placement considère les coordonnées initialisées de chaque sphère. Ces coordonnées sont recalculées en fonction des forces exercées sur une sphère et toutes les autres, en fonction des arêtes qui les relient. Par ces conditions, nous nous trouvons devant une impossibilité mathématique de positionner définitivement notre graphe. Par conséquent, nous décidons pour l'application des forces d'arrêter l'algorithme de placement lorsque toutes sphères auront été positionnées une première fois. Nous nous baserons alors sur les coordonnées obtenues à cet instant et en prenant en compte la valeur du lift attribuée aux différentes arêtes qui séparent les sphères entre elles pour l'évaluation des liens de corrélation.

Afin d'insérer la notion de longueur d'arc en fonction du lift, nous disposons dans les fonctions de répulsion (*Coulomb_fonction*) ou d'attraction (*Hooke_fonction*) de plusieurs constantes comme la charge éclectique, la constante de Coulomb ou de raideur du ressort *k*, ou encore la variable « L » qui prend en charge les étirements des ressorts. Il s'en suivra une déformation du graphe. Pour notre métaphore nous faisons intervenir le lift en utilisant la variable « L » et en attribuant à chaque ressort une valeur maximale pour son allongement.

- Pseudo-code de la fonction de répulsion (Coulomb) :

```
1. Coulomb_fonction (s1, s2) {
2. q1, q2 := 2.10-3 // constantes de charges éclectiques
3. force_C := (x = 0, y = 0, z = 0)
4. L := (x = 0, y = 0, z = 0)
5. modL := 0
6. Coulomb := 8.86 10-9 // constante de Coulomb
7. s_1 := (x1 = 0, y1 = 0, z1 = 0)
8. s_2 := (x2 = 0, y2 = 0, z2 = 0)
9.          L.x := s2.x2 – s1.x1
10.         L.y := s2.x2 – s1.x1
11.         L.z := s2.x2 – s1.x1
12.  modl := racine carrée (L.x²+L.y²+L.z²)
13.         If (Modl != 0)
14.         {
15.         force_C = (q1 *q2 * L) / (-4 *π * Coulomb * modl³)
16.         }
17.  return(force_C)
```

- Pseudo-code de la fonction d'attraction (Hook) :

```
1. Hooke_fonction (s_1, arc) {
2. force_H := (x = 0, y = 0, z = 0)
3. L := (x = 0, y = 0, z = 0)
4. modL := 0
5. kHooke := 0.2 // constantes de raideur du ressort
6.       s2 = edge.destination //le nœud lié à la sphère1
18.         L.x := s2.x2 – s1.x1
19.         L.y := s2.x2 – s1.x1
20.         L.z := s2.x2 – s1.x1
7.          modL := racine carrée (L.x²+L.y²+L.z²)
8.         if (modL!=0){
9.                 Force_H = kHooke*(modL - arc.(1/lift))*L / modL
10.         }
11.  return (force_H);
```

- Pseudo code de la fonction de placement :

```
1. var := 0
2. net_force :=  (x = 0, y = 0, z = 0)
3. n:= nbr_sphère
4. sphère := {x = xi, y = yi, z = zi};
5. arc := {idsphère, 1/lift},
6. Tant que var  < n :
7.  pour chaque autre_ sphère:
8.   net_force := net_force + Coulomb_fonction (var-sphère,
     autre_sphère)
9.          pour chaque arc lié à var-sphère :
10.          net_force := net_force  + Hooke_fonction (arc, var-sphère)
11.         prochain arc
12.  var := var + 1
13.  Fin de tant que
14.   net_force :=  (x = 0, y = 0, z = 0)
```

La FIG.35 montre le déplacement des sphères effectué par l'algorithme de placement. Ces déplacements sont effectués par rapport au centre du graphe (sphère jaune). Les sphères rouges représentent les positions initiales respectives de chaque sphère bleue ayant subit une transformation de coordonnées.

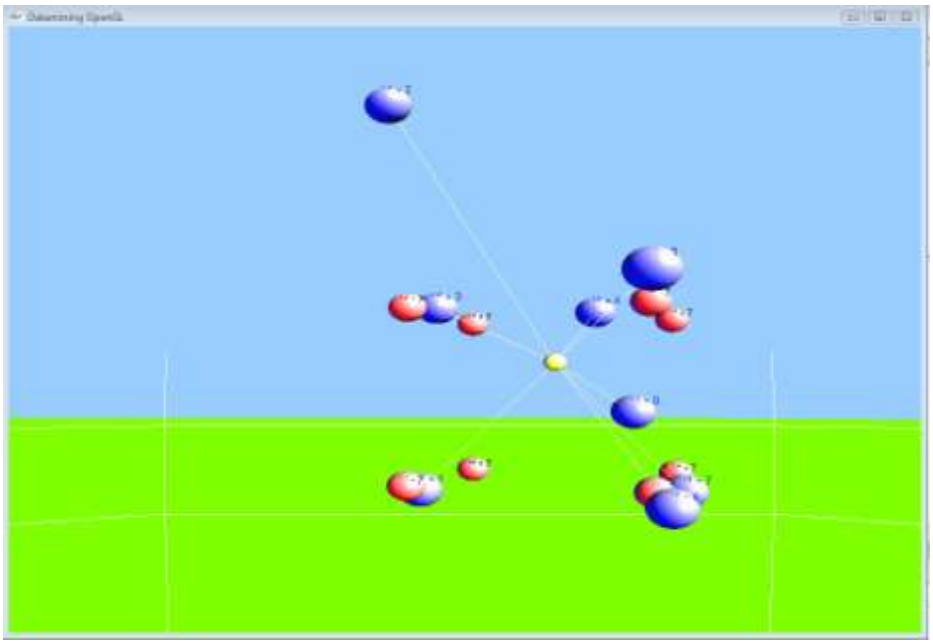
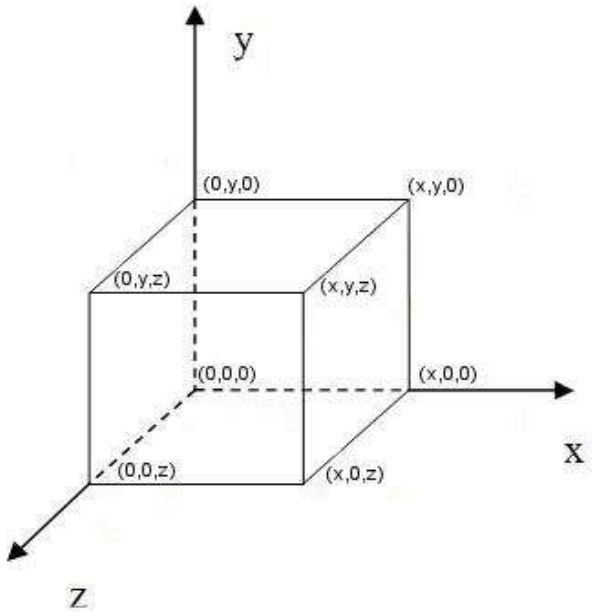


FIG.35 - Etirement du graphe

Remarquons que les sphères rouges sont positionnées sur les points de concours des arêtes d'un cube. En effet, l'initialisation des coordonnées (*x, y, z*) des sphères devra être réalisée de telle sorte que toutes les sphères soient équidistantes par rapport au centre du graphe qu'elles forment. Cette distance minimum servira alors de référence aux déplacements éventuels des sphères. Une comparaison significative des liens de corrélation par la distance inter-sphères sera donc possible suivant ce principe. Pour ce faire, nous limitons la longueur *k-itemsets* de la prémisse à huit, et positionnons les différents sommets du graphe sur les arêtes d'un cube.

Cette contrainte de limitation n'a pas d'influence sur l'applicabilité de notre prototype, les experts métiers préfèrent en générales un nombre d'attributs faible dans la prémisse d'une règle. Nous représentons ci-après la matrice pour les coordonnées (*x, y, z*) donnée par un code Gray[45]des huit sphères possibles pouvant être initialisées dans notre métaphore 3D.

[45] Le code Gray est fréquemment utilisé dans les capteurs angulaires ou de positionnement, mais aussi lorsque l'on désire une progression numérique binaire sans parasite transitoire.



q_0 :{0, 0, 0} ;
q_1 :{0, 0, z} ;
q_2 :{0, y, z} ;
q_3 :{0, y, 0} ;
q_4 :{x, y, 0} ;
q_5 :{x, y, z} ;
q_6 :{x, 0, z} ;
q_7 :{x, 0, 0} ;

FIG.36 - Coordonnées des points de concours des arêtes (repère OpenGL)

- L'indice de support

Nous utiliserons l'indice de support pour l'agencement et le positionnement des règles d'association dans l'arène 3D. La méthode utilise une matrice de transformation qui est appliquée sur les coordonnées initialisées (x, y, z) des sphères. Les coordonnées des points qui forment l'arène 3D en OpenGL serviront alors de référence à la matrice de transformation. Notons que les règles ayant les valeurs les plus élevées que procurent leurs supports seront positionnées respectivement aux positions les plus basses et avancées de l'arène.

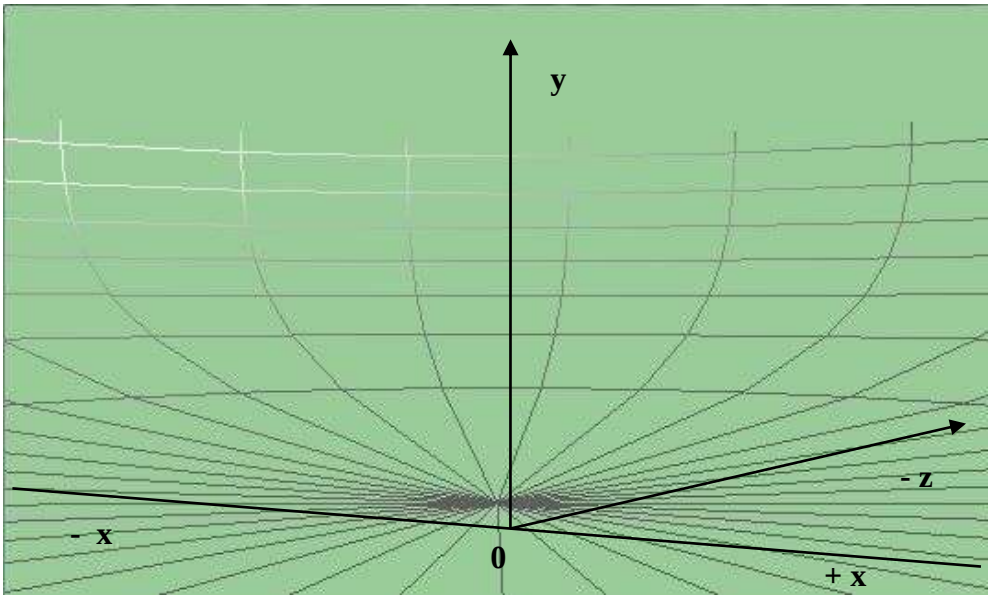


FIG.37 - Arène 3D (OpenGL)

Notons que le trièdre (*0, x, y, z*) dans une fenêtre OpenGL n'est pas orienté comme un repère orthonormé classique. La direction de l'axe (*z*) est placée sur la profondeur de la scène 3D. L'axe (*y*) exprime alors la hauteur de cette même scène. Par ailleurs, les lignes de l'arène 3D forment un nouveau repère dans lequel il est possible d'établir un système de coordonnées UTM[46].

A partir des angles qui définissent les lignes de latitude, la hauteur de l'arène est divisée en sept zones correspondantes à la projection des lignes de latitude dessinées par OpenGL sur l'axe [0, *y*]. Les règles d'association se juxtaposerons les unes à coté des autres suivant avec un pas fixe et incrémenté en suivant les valeurs de [$-x_{zone}$, $+x_{zone}$]. La première zone couverte par les règles extraites se trouvera donc au premier plan de la scène (au plus bas de l'arène donc disposent des supports les plus élevés).Les cordonnées correspondantes par projection en (*z*) seront données par l'équation du cercle de droite [o, x_{zone}].

Lorsque $x = x_{zone}$, la valeur de *x* est maximale. Le processus d'incrémentation recommence pour placer les règles dans la zone (segment de « y ») immédiatement supérieur. Par cette méthode, nous pouvons ajuster le pas entre chaque règle et augmenter la lisibilité de la représentation. La combinaison de la hauteur avec la profondeur pour le positionnement des règles assure la limitation des collisions entre les objets dans l'arène.

Afin d'assurer une certaine lisibilité des règles, une normalisation des différentes mesures d'intérêt sera nécessaire.

- L'indice de confiance

La notion de confiance accordée à une règle d'association sera représentée par la distance qui sépare le graphe de prémisse avec le graphe de conclusion (dans notre cas un seul *item*).

[46] U.T.M. est l'acronyme de l'anglais *Universal Transvers Mercator*. C'est une méthode de découpage de la terre pour situer un point avec précision.

- Récapitulatif des mesures de qualité et encodages graphiques

Indices de qualité	Encodages graphiques
1 / Lift	*Distance entre les différents items de la prémisse.*
Support	*Un support élevé positionne une règle au plus bas de l'arène 3D.*
Confiance	*Distance (longueur de l'arc) entre le graphe de prémisse et l'item de conclusion.*
Le gain informationnel de Freitas	*Diamètre et volume des sphères représentant la molécule.*

TAB.15 - Correspondance graphique (objets/mesures)

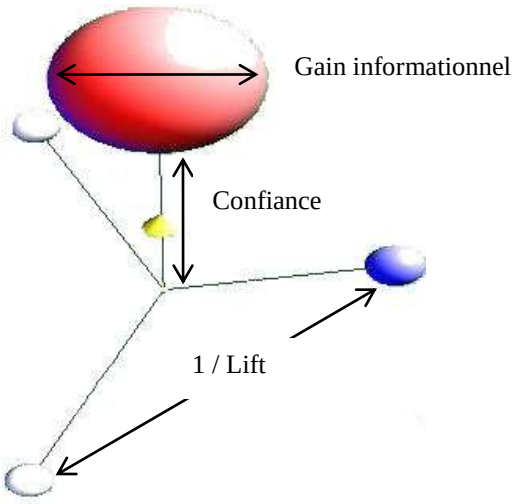


FIG.38 - Encodage graphique des mesures d'intérêt

Chapitre 4

Réalisation

4.1 Présentation

La réalisation se décompose en deux étapes :

- Une première partie présente les différentes séquences de requêtes pour extraire les règles en parcourant le contexte de données.

- La deuxième partie traite de l'étude et de l'expérimentation de la métaphore dans une arène 3D.

Dans ce chapitre, une réflexion spécifique est portée sur la représentation graphique des liens de corrélation existants entre les différents *items* de la prémisse. Nous rappelons que ces liens sont donnés par le lift. Nous limiterons le nombre de ses *items* à huit, (contrainte vue précédemment). L'extraction des règles est supervisée par l'utilisateur qui sélectionne les attributs de la prémisse. Les algorithmes d'extraction assujettis à cette contrainte limitent drastiquement l'espace de recherche. Les sous-ensembles de règles obtenus sont visualisés directement dans l'arène 3D. L'application utilise en entrée un fichier de données codé au format CSV (tableau transaction/attributs, séparateur « ; »).

4.2 Organisation

4.2.1 Dates clés

- 02/11/2009 : début du stage,
- 18/01/2010 : validation du sujet,
- 15/03/2010 : présentation du serveur de production de règles,
- 15/05/2010 : présentation d'un prototype de métaphore de règles,
- 15/07/2010 : validation des algorithmes de calcul,
- 18/02/2011 : soutenance.

4.2.2 Planification des tâches

La période de bibliographie prévoyait une durée de trois mois. À cette période, s'intègre la proposition et la rédaction du sujet de mémoire. Côté programmation, un temps d'adaptation aux outils de développement et au langage a été nécessaire, ce temps ne figure pas (FIG.39) puisqu'il est intégré dans l'étude de la technologie OpenGL. L'extraction des premières règles d'association à partir du serveur fût rapide. Cependant le développement de la métaphore à été plus long que prévu. Son développement à nécessité d'effectuer de nombreux essais pour trouver une solution cohérente. La phase d'intégration et de débogage est partagée pour les deux modules principaux de programmation (serveur de production de règles, métaphore3D). Le temps restant du stage n'étant pas suffisant pour aborder et implémenter les techniques d'interaction, nous avons donc expérimenté le système sur des données réelles fournies par Nantes habitat.

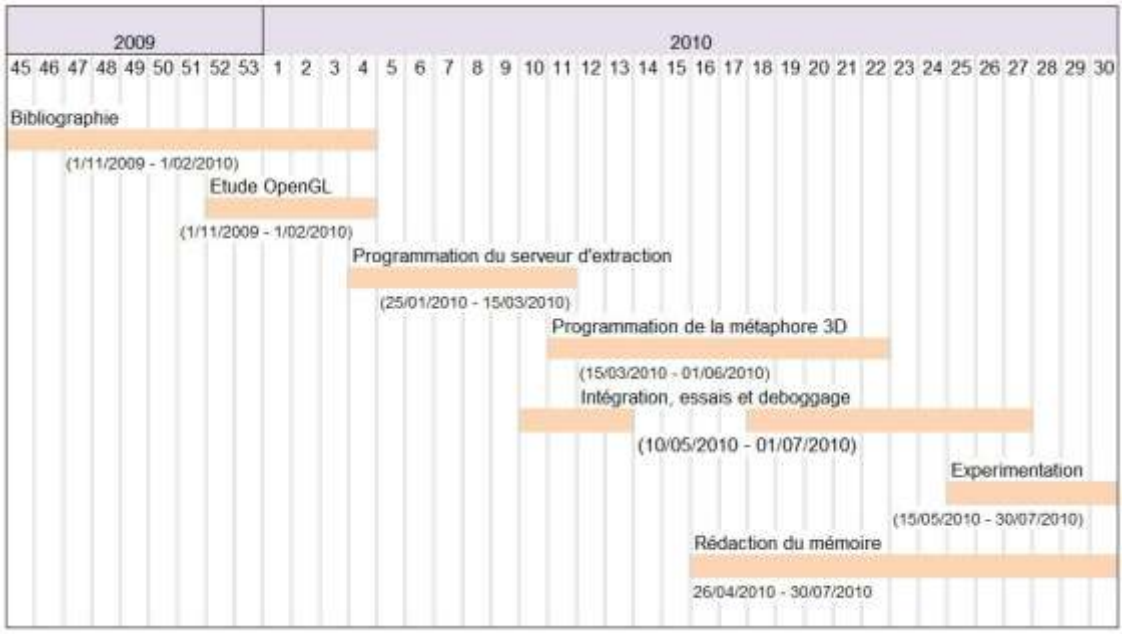


FIG.39 - Diagramme de Gantt des tâches du projet

4.3 Contexte technique

Le processus de fouille de règles s'articule autour de trois composantes connectées et évoluant en fonction des requêtes de l'utilisateur (FIG.40) :

- une base de données qui contient les données de l'étude et les tables de travail pour optimiser le temps de réponse des requêtes,

- une heuristique qui calcule les sous-ensembles de règles extraits suivant les souhaits de l'utilisateur. Cette heuristique travaille de façon locale sur les données. Les mesures associées ont été présentées au chapitre précédent (voir TAB.15).

- une interface de visualisation et une métaphore de règles.

Le programme d'extraction de règles d'association est écrit en C/C++, il anime un jeu de requêtes SQL pour interroger le contexte de données. L'initialisation se fait par une interface en lignes de commande. Lorsque l'algorithme d'extraction arrive en fin de processus, le programme ouvre une fenêtre graphique OpenGL. Le développement s'appuie sur plusieurs librairies disponibles.

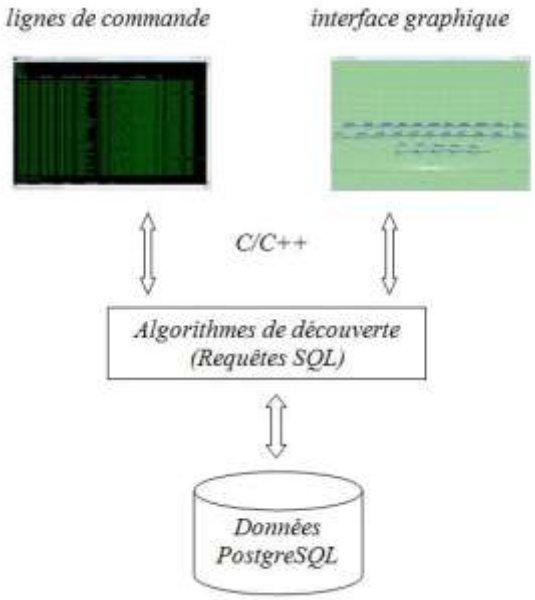


FIG.40 - Architecture du processus de fouille

4.4 Normes et outils

L'application est spécifiée en UML à l'aide du logiciel de modélisation Modélio[47]. La norme UML (*Unified Modeling Langage*) est une méthode de description sous forme de graphes des données et des traitements. C'est une formalisation de la modélisation objet[48]. Elle facilite la manipulation de concepts abstraits liés à cette méthodologie (héritage, polymorphisme.etc.) et permet par son aspect graphique et par le large consensus réalisé autour de ses méthodes[49] un échange aisé entre les différents acteurs du projet. L'utilisation d'UML s'avère particulièrement utile pour les développements de projet itératif. UML facilite la communication et les modifications d'éléments existants grâce aux caractéristiques « objet » qui limitent le couplage entre les différents modules de l'application.

[47] http://www.modeliosoft.com/
[48] http ://fr.wikipedia.org/wiki/Unified_Modeling_Language
[49] http : //uml.free.fr/

4.5 Spécification

4.5.1 Diagramme de classe simplifié

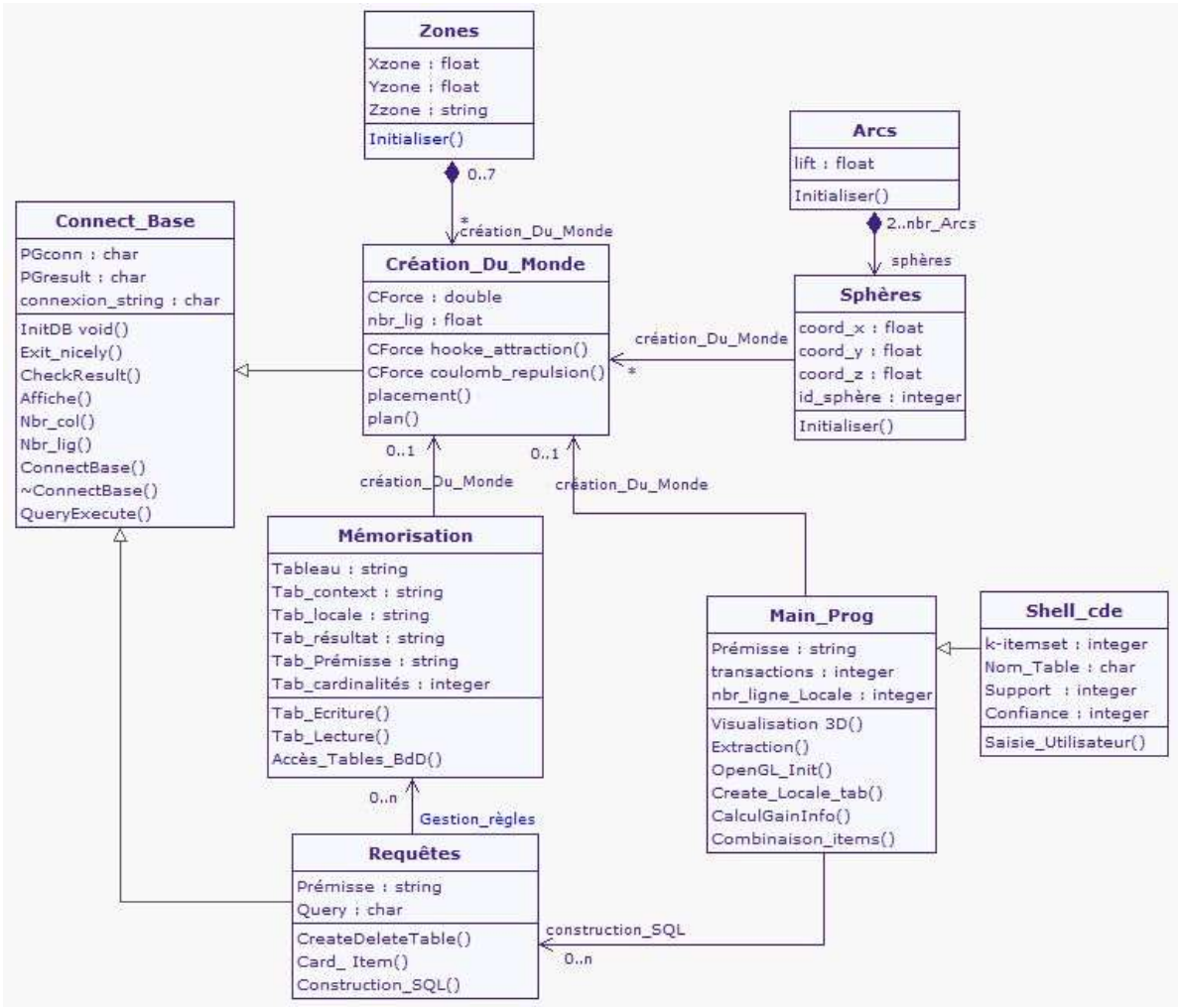


FIG.41 - Diagramme de classes simplifié

4.5.2 Description des classes principales :

- **Shell_cde** : initialisation de la primitive d'extraction, sélection de la longueur de la prémisse et identification des *items*. Choix des seuils pour les indices de confiance et de support

    - Saisie_Utilisateur : Cette méthode ne prend pas de valeur en entrée mais ouvre une interface de saisie et construit les couples variable/attribut. Ces couples sont ensuite enregistrés dans un tableau à deux dimensions comportant un champ pour l'*item* et un champ pour sa variable correspondante.

- Pseudo-code pour la saisie des items :

```
1.k_itemset := nbr_items
2.n := 0
3.var [] := Ø
4.att [] := Ø
5.Prémisse [] := Ø
6.Tant que n < k_itemset
7.   Faire
8.    att = item
9.    var = n°colonne
10.              Identification_variable (n°colonne)
11.              Prémisse [] := (variable/item)
12.         Fin de faire
13.       Fin de tant que
14.       Return (Prémisse)
```

- **Main_Prog** : Le point d'entrée du programme se trouve dans cette classe. Les méthodes qu'elle regroupe assurent la saisie des *items* pour établir une condition de prémisse ainsi que le chargement des tables de travail intermédiaires. Les autres méthodes qui figurent dans cette classe gèrent les fonctions de visualisation d'OpenGL. Description des méthodes principales :

    - Extraction : Lorsque l'utilisateur valide sa sélection (*k_itemset* et attributs), une requête de type (`INSERT INTO Tab_locale SELECT * FROM Prémisse`) est exécutée sur le serveur. La table de contexte locale est alors renseignée. Cette méthode prend en entrée :

        - les valeurs des seuils de support et de confiance (sup et conf),
        - la table locale de données,
        - le nombre de lignes et de colonnes de la table locale,
        - le nombre de transaction « t » du contexte global,
        - la chaine de caractères « Prémisse » qui contient les correspondances variable/attribut de la prémisse.

      L'algorithme d'extraction effectue alors la recherche des règles en mode local et en fonction des seuils de support et de confiance. La fonction « Lecture » est une fonction de recherche d'intitulé pour la variable par rapport à l'attribut que l'on considère pour la conclusion.

      - Pseudo-code pour extraire les règles :

```
1. lig := nbr_lignes (Tab_locale)
2. col := nbr_colonnes (Tab_locale)
3. t := nbr_transaction (Tab_générale)
4. i, j := 0
5. sup, conf := 0
6. var [] := Ø
7. att [] := Ø
8. GI := 0
9. cardL := 0 //cardinalité des items table locale
10. cardG := 0 //cardinalité des items table globale
11.    Tant que (i < lig)
```

```
12.      Tant que (j <col)
13.         Faire
14.            Lecture (Tab_locale)
15.             att [] := item_conclusion
16.             var [] := variable
17.          Tant que (att ∉ Prémisse [])
18. cardL := select count(*)from Tab_locale WHERE
    var = att
19. cardG := select count(*)from Tab_globale WHERE
    var = att
20.             sup := (cardL /t)
21.             conf := (cardL /lig)
22.
23.           Si (sup > seuil_sup et conf > seuil_conf)
24. GI := CalculGainInfo (att, cardG, cardL, lig, t)
25.    Insérer_règle (Prémisse, sup, conf, GI)
26.            Fin de si
27.        Fin de tant que
28.       Fin de Faire
29.      Fin de tant que
30.    Fin de tant que
```

- CalculGainInfo : Cette méthode est appelée au cours de l'extraction des règles et prend en entrée :

  - la variable de l'*item* (att) de la conclusion de la règle,
  - la cardinalité de (att) dans le contexte général (cardG),
  - le nombre de transactions du contexte général (t).

  - Pseudo-code pour le calcul du gain informationnel (*item* de la conclusion) :

```
1. var []:= variable_conclusion
2. att [] := item_conclusion
3. nbr_att := nbr_item_conclusion
4. n := 0
5. Gi := 0
6. ¬Gi := 0
7. t := nbr_transaction (Tab_générale)
8. InfoGain := 0
9. Tant que n < nbr_att
10.    Gi := SELECT variable, COUNT (*) FROM
   Tab_générale WHERE variable =
   item_conclusion
11.    ¬Gi := SELECT variable, COUNT (*) FROM
   Tab_générale WHERE variable = ¬item_conclusion
12.    InfoGain := -((Gi /t)*log10 (Gi/t)+
   (¬Gi/t)*log10 (¬Gi/t))
13.    InfoGainConclusion:= InfoGainConclusion +
   InfoGain
14.    Fin de Tant que
15.    Return (InfoGainConclusion)
```

  - Pseudo-code pour le calcul du gain informationnel des attributs de la prémisse :

```
1. var []:= variable_conclusion
2. att [] := item_conclusion
3. GI_conclu := InfoGainConclusion
4. k_itemset
5. InfoGain := 0
```

```
6. nbr := 0
7. varatt [] := variable_prémisse
8. att_p [] := item_prémisse
9. Ai := 0
10.    ¬Ai := 0
11.    Gi|Ai := 0
12.    ¬Gi|Ai := 0
13.    Tant que nbr < k_itemset
14.       Faire
15.    Ai := SELECT varatt, COUNT (*) FROM
   Tab_générale WHERE varatt = att_p [nbr]
16.    ¬Ai := SELECT varatt, COUNT (*) FROM
   Tab_générale WHERE varatt = ¬att_p [nbr]
16.    Gi|Ai := SELECT varatt, COUNT (*) FROM
   Tab_générale WHERE varatt = att_p [nbr] GROUP BY
   att
17.    ¬Gi|Ai := SELECT varatt, COUNT (*) FROM
   Tab_générale WHERE varatt = att_p [nbr] GROUP BY
   ¬att
18.    Gi|¬Ai := SELECT varatt, COUNT (*) FROM
   Tab_générale WHERE varatt = ¬att_p [nbr] GROUP
   BY att
19.    Fin de Faire
20.    InfoGain = InfoGain +
    Ai * (-((Gi|Ai)*log(Gi|Ai)+ (¬Gi|Ai)*
   log(¬Gi|Ai))+
   ¬Ai * (-( Gi|¬Ai)*log(Gi|¬Ai)+ (¬Gi|¬Ai)*
   log(¬Gi|¬Ai))
21.    Fin de Tant que
22.    Return (InfoGain)
```

- Combinaison_items : Cette méthode prend en entrée la chaine de caractère « Prémisse » et sa longueur (k_itemset). Une combinaison deux à deux des attributs est alors effectuée pour l'évaluation de la mesure du lift existant entre chacun des attributs contenu dans la chaine « Prémisse ».

- Pseudo-code pour l'évaluation du lift entre les items deux à deux:

```
1. k_itemset := nbr_items dans la prémisse
2. Prémisse [item/variable]
3. t := nbr_transaction (Tab_générale)
4. var := 0
5. lift := 0
6. Card := 0
7. Card1 :=  0
8. Cardd2 := 0
9. item1 [] := 0
10.       item2 [] := 0
11.       variable1 [] := 0
12.       variable2 [] := 0
13.       QUERY [] := 0
14.       Tant que var < k_itemset
15.       Faire
16.         Lecture (Tab_générale)
17.         item1 := Prémisse [var]
18.         variable1 := Prémisse [var]
19.         Card1 := SELECT COUNT(*)FROM  Tab_générale
                  WHERE variable1= 'item1'
20.               Tant que var < k_itemset
```

```
21.              Faire
22.                Lecture (Tab_générale)
23.                 item2 := Prémisse [var+1]
24.                 variable2 := Prémisse [var+1]
25.          Card2 := SELECT COUNT(*) FROM Tab_générale
                  WHERE variable2 = 'item2'
26.          Card := SELECT item1, COUNT (*) FROM
   Tab_générale WHERE variable2 = 'item2'
   GROUP BY item1
27.       lift := ((Card/t)/((Card1/t)*(Card2/t)));
28.       QUERY := INSERT INTO Tab_prémisse (item1,
   item2, lift)
29.       Envoyer(QUERY)
30.              Fin de Faire
31.              Fin de Tant que
32.       Fin de Faire
33.       Fin de Tant que
```

- Chargement tables : Les données en sortie de l'algorithme sont écrites dans les tables de la base de données (tables de cardinalité des différents attributs et la table résultat pour l'écriture des règles). On effectue ensuite une opération de réécriture des données à partir de la base de données (table de résultat) vers un tableau en C pour que la fonction de visualisation puisse accéder aux données dans de bonnes conditions. Des essais ont été effectués en interrogeant directement la base de données, mais le nombre d'itérations induit par le nombre de données brutes demande à l'algorithme d'effectuer un trop rand nombre de requêtes sur la base de données et diminue considérablement l'efficacité du système.

- Visualisation 3D : la visualisation se charge de la connexion à la base de données, du chargement du tableau en C et l'appel des fonctions pour la gestion de l'affichage OpenGL.

- **Mémorisation :** Au démarrage si la connexion à la base de données est effective, le programme assure la création de quatre tables de travail :

  - Tab_prémisse : cette table contient les différentes associations d'*items* de la prémisse issus de la combinaison deux à deux entre eux (voir FIG.31). Elle contient trois variables (syntaxe de règles, indice de support, de confiance et le lift). Seul le lift sera utilisé pour la représentation graphique.

  - Tab_locale : les attributs par variables et correspondant aux conditions envoyées par la requête définie suivant les choix de l'utilisateur sont écris dans cette nouvelle table. Les traitements pour le calcul des indices s'effectuent alors à partir de cette table en parallèle de la table contenant le contexte général.

  - Tab_cardinalité : les cardinalités des attributs contenus dans les tables de contexte général ou local sont données par les requêtes figurant dans les pseudo-codes précédente (ex : lignes 18 et 19

extraction des règles). Lorsque la cardinalité d'un attribut est calculée, si elle n'existe pas déjà dans la table par une itération précédente, elle y est insérée. Cette table permet d'alléger les requêtes après une phase d'initialisation. La charge de travail de la base de données se limitera ensuite à tester la présence du couple attribut/variable dans la table. Ce principe améliore le temps de réponse du système client/serveur.

- Tab_résultat : l'algorithme d'extraction écrit les règles d'association dans la table résultat. Les champs contiennent la syntaxe de la règle, le gain informationnel de chacun des attributs de l'*itemset*, le support, la confiance et le lift.

	oid	premisse text	conclusion text	gain_info_pre text	gain_info_cor text	support double precis	confiance double precis	lift double precis
1	5389946	outlook (sunny)	water(0.2)	0.8095|-814158	1.06758	5	25	166.67
2	5389947	outlook (sunny)	rain(hight)	0.1564|-224078	0.46343	5	25	83.33
3	5389948	outlook (sunny)	water(0.6)	0.9547|-575362	1.06758	5	25	500
4	5389949	outlook (sunny)	rain(low)	0.1678|-218745	0.46343	10	50	111.11
5	5389950	outlook (sunny)	water(0.4)	0.8487|-128786	1.06758	5	25	250
6	5389951	outlook (sunny)	water(0.25)	0.8095|-190320	1.06758	5	25	166.67
7	5389952	outlook (sunny)	rain(medium)	0.1539|-217894	0.46343	5	25	100
*								

FIG.42 - Table de résultats (*pgAdmin*)

Nous montrons (FIG.42) l'ensemble de règles obtenues à partir du modèle de données (FIG.25). Notons que la règle retenue en exemple que nous rappelons : (Outlook = sunny, Temp = hot, Windy = yes) → (Rain = low) et ses mesures de qualité associées y figurent.

- **Connect_Base** : elle intègre la librairie de *PostgreSQL* fournie pour le développement d'applications en C et gère la connexion et les requêtes effectuées sur la base de données. Les principales méthodes implémentées vérifient la syntaxe des requêtes ou informent sur le nombre de lignes et de colonnes d'une table en affichant ses valeurs.

- **Construction SQL** : Cette classe hérite des propriété de connexion de la classe *Connect_Base* ». Ses fonctionnalités créent des tables de travail et séquencent les différentes requêtes à appliquer pour l'extraction des règles d'association. Les mesures de qualités sont alors calculées comme vu précédemment par des méthodes indépendantes et à l'aide de requêtes dédiées. Ces méthodes parcourent le contexte global ou local des données.

- **Création_Du_Monde** : une relation d'agrégation lie les classes « *Sphères* » et « *Arcs* ». Un arc est initialisé si au moins deux sphères sont initialisées. L'algorithme de placement fait appel à ces deux classes pour l'initialisation et l'identification des différentes sphères et arcs qui forment le graphe de prémisse d'une règle d'association. Les positions des sphères entre elles sont définies par les fonctions d'attraction et de répulsion. Les coordonnées

initialisées des sphères subissent alors les transformations de coordonnées définies par l'algorithme de placement vu § 3.3.5 du chapitre 3.

- **Zones :** la méthode Initialiser () de cette classe prend en entrée les coordonnées initiales des différentes sphères. Ces coordonnées sont ensuite combinées à une matrice de correspondance pour positionner les règles dans l'arène. Les valeurs de la matrice sont définies par rapport à l'échelle du dessin OpenGL. Nous allons détailler cette méthode. Nous commençons par l'initialisation des différentes zones avec les valeurs expérimentales données par le dessin OpenGL de la scène. Nous initialisons également les coordonnées de chaque sphère qui représentent les attributs.

  - Matrice de coordonnées des zones

```
1. Zone [0] := {ref, 0,-19,-7}
2. Zone [1] := {ref, 0,-18,-23}
3. Zone [2] := {ref, 0,-16,-38}
4. Zone [3] := {ref, 0,-13,-50}
5. Zone [4] := {ref, 0,-9,-260}
6. Zone [5] := {ref, 0,-5,-68}
7. Zone [6] := {ref, 0,0.7,-73}
```

  - Matrice de coordonnées des sphères

1. Sphère [0] := {id_0, x_0, y_0, z_0}
2. Sphère [0] := {id_1, x_1, y_1, z_1}
3. Sphère [0] := {id_2, x_2, y_2, z_2}
4. Sphère [0] := {id_3, x_3, y_3, z_3}
5. Sphère [0] := {id_4, x_4, y_4, z_4}
6. Sphère [0] := {id_5, x_5, y_5, z_5}
7. Sphère [0] := {id_6, x_6, y_6, z_6}
8. Sphère [0] := {id_7, x_7, y_7, z_7}

  - Pseudo-code de l'algorithme de positionnement

1. i := 0
2. j := 0
3. var := nbr_lignes (Tab_résultat)
4. nbr_sphère := 8
5. cir := 0
6. Sphère [i] := {$id_{sphère}$[i], $x_{sphère}$[i], $y_{sphère}$[i], $z_{sphère}$[i]}
7. Zone [j] := {ref_{zone}[j], x_{zone}[j], y_{zone}[j], z_{zone}[j]}
8. ref_{zone}[j] := nbr_sphère
9. espace := - (z_{zone}[j] *2)/ (z_{zone}[j])
10. rayon := z_{zone}[j] + espace
11. pas := rayon + espace
12. Tant que pas < var
13. si (pas > ref_{zone}[j] - var)
14. Faire
15. j := j + 1
16. ref_{zone}[j]:= -(z_{zone}[j]*2)
    espace := espace +
    (ref_{zone}[j]*2/ ref_{zone}[j])
17. rayon := z_{zone}[j]
18. pas := rayon + espace
19. Fin de Faire
20. Sinon
21. pas := pas +espace
22. cir := $rayon^2 - pas^2$
23. si (cir < 0)
24. Faire

```
25.                                cir := -cir
26.                                Fin de Faire
27.                          Fin de si
28.                          cir := racine carrée (cir)
29.                  Tant que i < nbr_sphère
30.                    Faire
31.                          x_sphère[i] := x_sphère[i] + pas
32.                          y_sphère[i] := y_sphère[i] +
33.                          y_zone[j] + z_sphère[i]
34.                          z_zone[j] := z_zone[j] - cir
35.                    Fin de Faire
36.                  Fin de tant que
37.                  Fin de si
38.          Fin de Tant que
```

La matrice définie sept zones de coordonnées constantes pour les coordonnées *y* et *z* de chaque sphère. Elle assure la corrélation entre la hauteur et la profondeur pour la position de chaque règle dans l'arène 3D, (FIG.43).

Les tableaux en C sont triés de façon croissante par rapport à la mesure du support. Les itérations pour l'algorithme de positionnement commencent par la zone [0] donc à la position la plus basse et avancée dans l'arène.
Ces règles possèdent les indices de support les plus élevés. Les valeurs de *x* nous assurent le balayage de chaque zone de [-xzone, +xzone]. Cette valeur est incrémentée entre chaque règle d'association avec un pas variable en fonction de la zone considérée. L'algorithme de positionnement prend en charge toute les règles de la table, cependant au-delà de la zone [6] et pour le dernier pas, c'est-à-dire une centaine de règles, les règles ne seront plus représentées (FIG.44). Le système de visualisation lit le tableau en boucle et reprend les opérations dans l'ordre d'exécution suivant :

- initialisation et transformation des coordonnées de chaque sphère avec la matrice de positionnement,
- initialisation des arcs qui séparent les sphères,
- transformation des coordonnées des sphères par l'algorithme de placement (Coulomb et Hooke).

Il sera possible pour des besoins futurs de recharger dynamiquement le tableau à la demande de l'utilisateur et pour une nouvelle primitive d'extraction. Dans ce cas, le système devra être synchronisé par un mécanisme de gestion de sémaphore (*thread*) pour le partage de l'accès au tableau entre la visualisation pour la lecture et l'extraction des règles pour l'écriture.

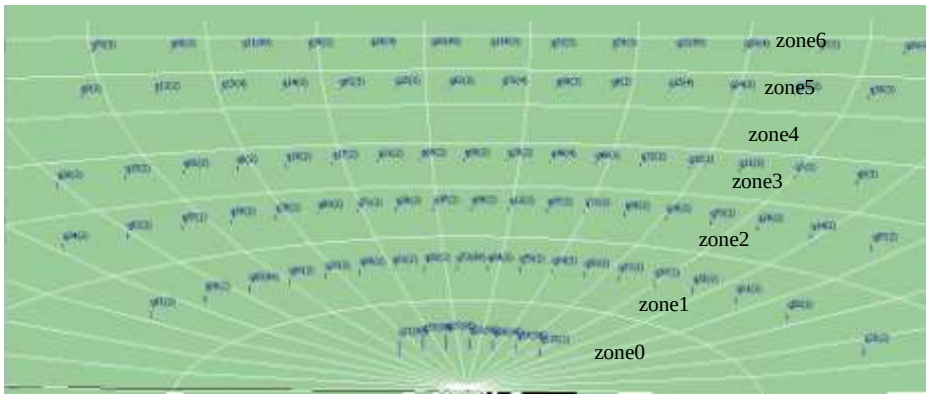


FIG.43 - Calcul des coordonnées dans l'arène

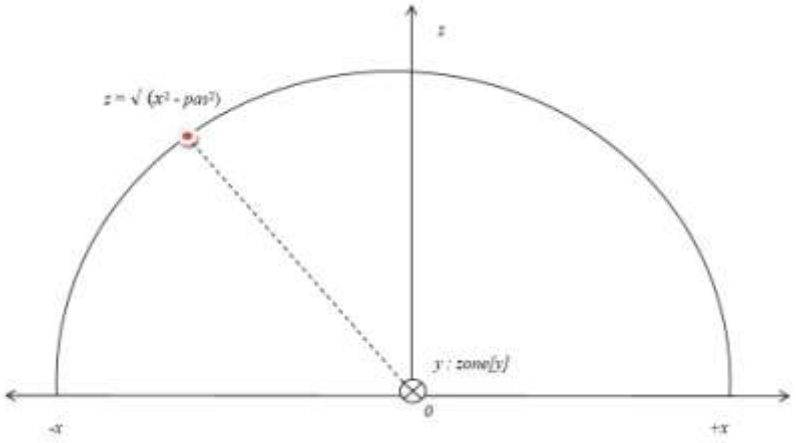


FIG.44 - Positionnement des règles dans l'arène

4.5.3 Interface utilisateur

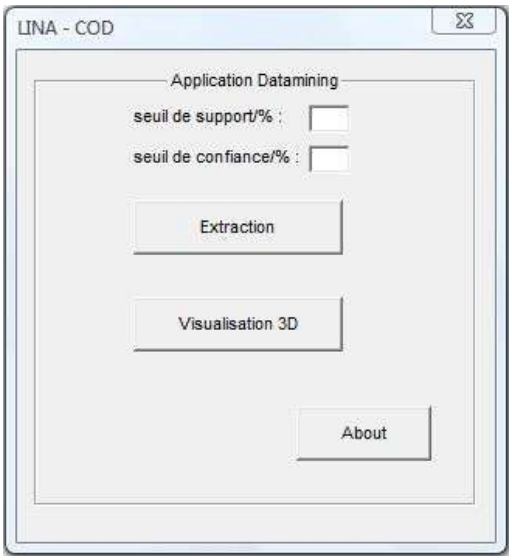


FIG.45 - Interface utilisateur

Le programme démarre par l'ouverture d'uns boite de dialogue (FIG.45). Elle contient deux boutons d'action. Ces boutons séparent la partie extraction de la partie visualisation. L'interface permet également de renseigner les seuils des indices de support et de confiance nécessaires à l'extraction des règles. La boite de dialogue est programmée avec l'API de MS-Windows.

4.5.4 Diagramme d'activité : interface

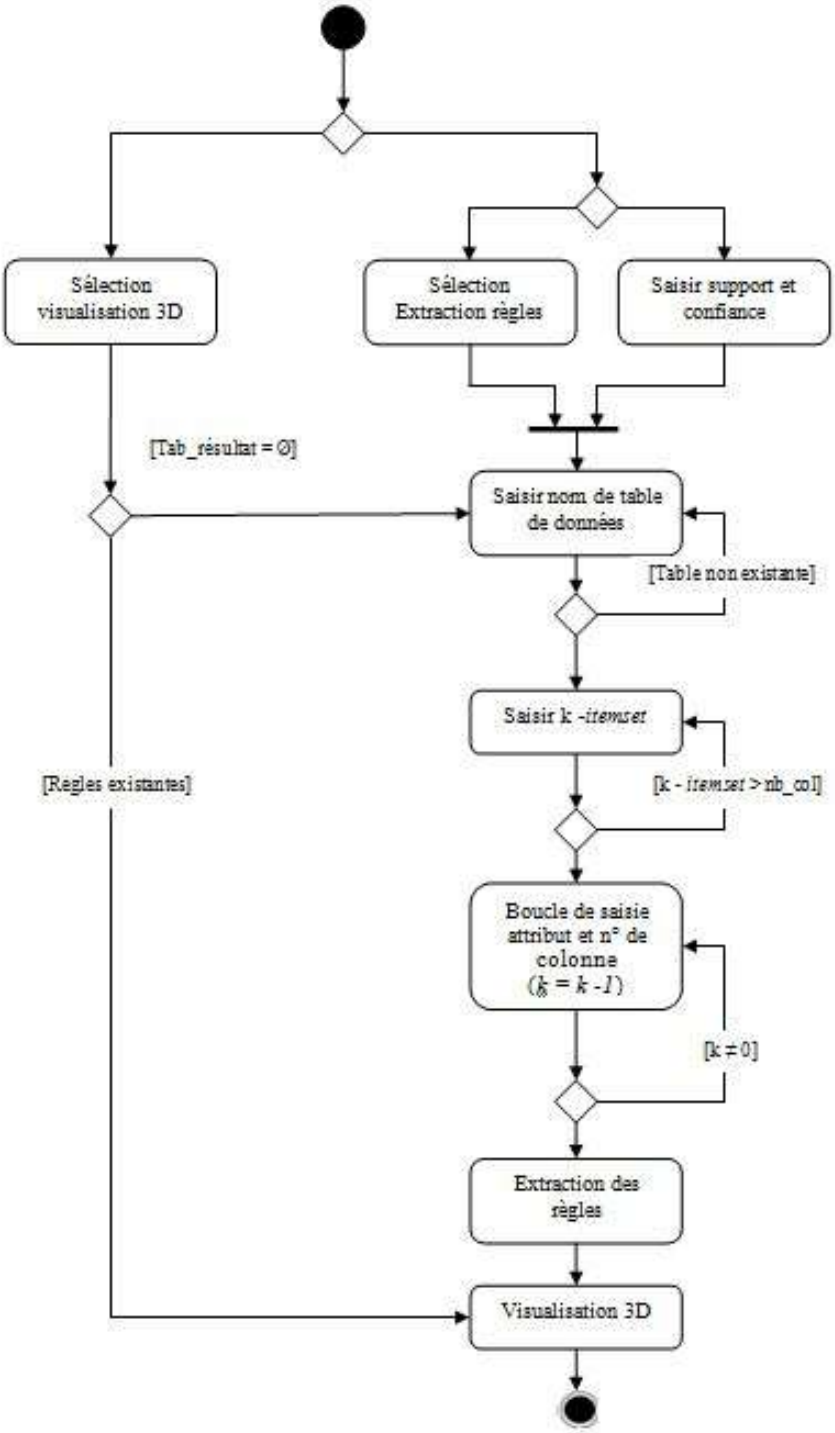


FIG.46 - Diagramme d'activité

Au démarrage, le programme offre le choix à l'utilisateur d'activer l'une ou l'autre des deux fonctionnalités principales :

- La fonction d'extraction de règles : le système se connecte à la base de données et ouvre une console (Shell). La fonction prend en entrées le nombre des attributs pour construire les règles, la syntaxe de chaque attribut et leur numéro de colonne respectif correspondant à la variable.
- la fonction de visualisation : le système se connecte à la base de données par l'objet de connexion. Cette fonction prend en entrée les données contenues dans la table de résultat et intègre les fonctionnalités de gestion des piles matricielles spécifiques à OpenGL.

4.5.5 Diagramme d'activité : extraction des règles (*class* Main_Prog)

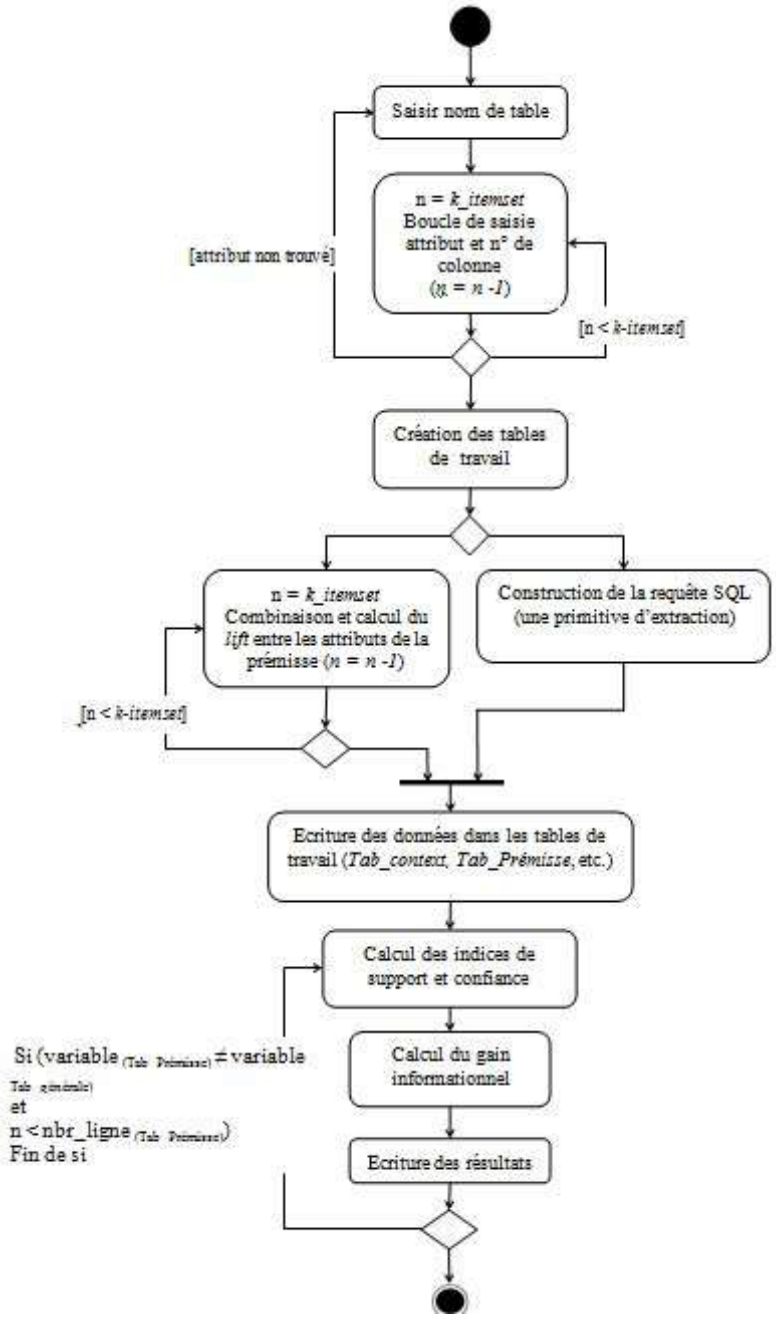


FIG.47 – Diagramme d'activité (extraction des règles)

4.5.6 Comparaison des résultats d'extraction avec Tanagra

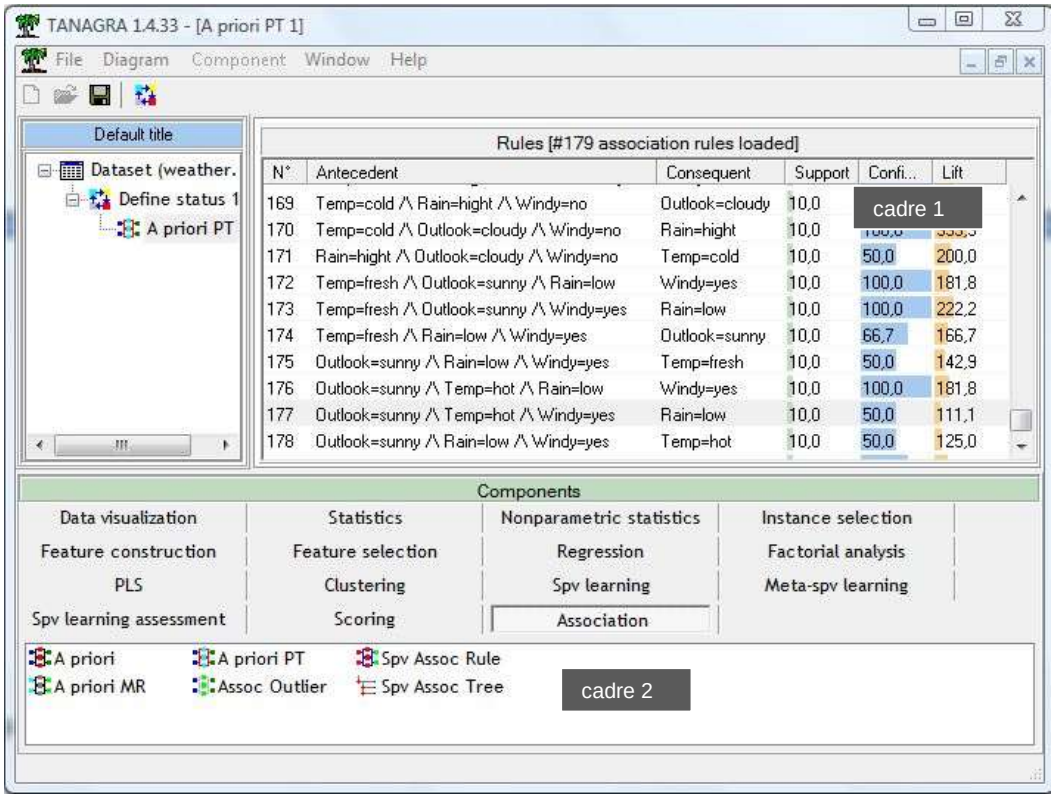


FIG.48 - Extraction avec *Tanagra*

La FIG.48 montre les résultats (cadre_1) obtenus en utilisant un algorithme exhaustif de règles d'association. Cet algorithme écrit par Christian Borgelt[50] est implémenté dans les fonctionnalités de *Tanagra* (cadre_ 2). Il nous intéresse dans sa particularité d'extraire des règles à conclusion simple et de pouvoir ainsi valider nos résultats. Son utilisation dans *Tanagra* permet de fixer des pré-conditions d'extraction et de renseigner les seuils de support et de confiance tout comme le fait notre application. Notre règle en référence est décrite à la ligne 177 (cadre 1).

[50] http://www.borgelt.net/pub2010.html

4.6 Placement du graphe de prémisse

4.6.1 Initialisation de l'algorithme de placement:

Les coordonnées initiales des différentes sphères sont placées dans la matrice de coordonnées suivante.

q_0 :{0, 0, 0} ;
q_1 :{0, 0, 1} ;
q_2 :{0, 1, 1} ;
q_3 :{0, 1, 0} ;
q_4 :{1, 1, 0} ;
q_5 :{1, 1, 1} ;
q_6 :{1, 0, 1} ;
q_7 :{1, 0, 0} ;

L'algorithme de placement utilise simultanément les fonctions d'attraction et de répulsion. A chacune de ses itérations, la matrice de coordonnées est modifiée. Il s'en suit la définition d'une nouvelle position pour chacune des sphères dans la scène 3D. Lorsque toutes les sphères sont positionnées, l'algorithme reprend les itérations en commençant par la première sphère de la règle suivante. Il réalise ainsi les mêmes opérations pour toutes les règles à afficher.

Les déformations du graphe sont engendrées par l'ensemble des forces qui y sont appliquée (voir FIG.37). L'algorithme effectue alors le calcul des forces d'attraction et de répulsion en initialisant une charge électrique (poids) à chacune des sphères (sommets du graphe) et en attribuant un allongement maximum des ressorts qui les relient entre elles. Cet allongement est élaboré à partir de la mesure du lift. Les nouvelles positions des sphères sont ainsi définies par l'ensemble des forces qui s'exercent à l'intérieur du graphe. Le parcours du graphe complet par l'algorithme ne nous permet pas de fixer les coordonnées dans l'espace 3D. Pour le placement des sphères, nous forçons donc l'arrêt des itérations lorsque tous les arcs qui les relient entre elles, ont été pris en compte.

4.6.2 Echelle graphique

Nous choisissons d'attribuer une échelle unitaire pour les cordonnées d'initialisation des sphères dans l'arène. Ce choix facilite l'interprétation expérimentale des liens de corrélation (lift) par rapport aux distances qui les séparent. Cette échelle assure également la lisibilité des mesures au sein des règles lorsqu'elles celles-ci sont positionnées dans l'arène 3D (FIG.54).

La charge électrique dans le cadre de cette étude reste constante et identique pour chacune des sphères. Nous limitons les forces de répulsion avec des charges électriques faibles. Une charge électrique élevée a pour effet de faire sortir les sphères de l'arène. Nous attribuons également une valeur expérimentale à la constante de Coulomb (voir §3.3.5) pour les besoins de la métaphore. Les constantes électriques nous serviront donc à gérer le placement du graphe pour chacune des règles en concordance avec l'échelle graphique de la fenêtre OpenGL.

4.6.3 Corrélations graphiques ente le lift et les distances inter-sphères

Afin d'obtenir une indication de corrélation cohérente avec les distances qui séparent les sphères et la valeur de la mesure du lift. L'initialisation des coordonnées est une phase prépondérante. Les coordonnées de référence pour calculer les distances dans notre métaphore sont les coordonnées des points de concours des arêtes d'un cube. Le cube offre la possibilité d'initialiser huit sphères positionnées à égales distances par rapport à son centre de gravité. Dans l'exemple (FIG.49), nous montrons un nombre de cinq sphères qui nécessitent pour l'algorithme, d'initialiser dix arcs afin de simuler les ressorts (voir § 3.3.5).

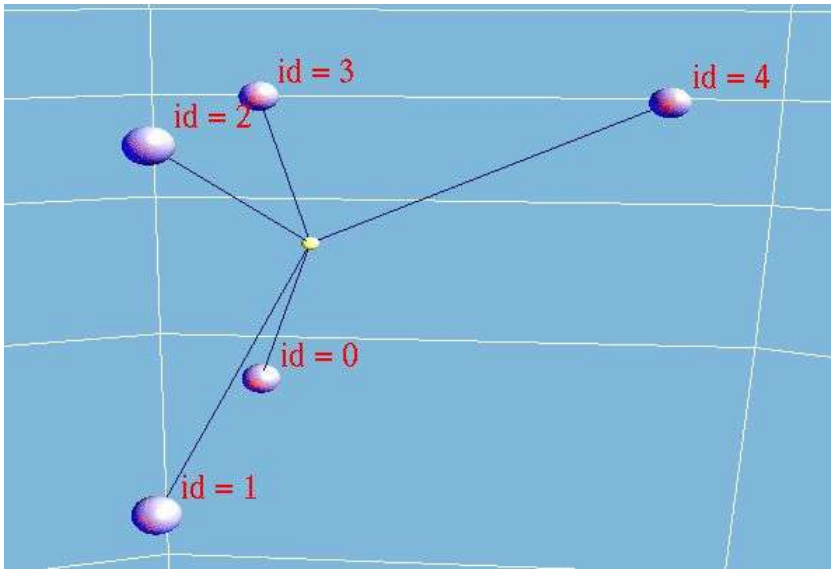


FIG.49 - Initialisation des coordonnées

Cependant, la méthode de placement doit prendre en compte les différences existantes entre les longueurs des arcs positionnés :

- sur les diagonales du cube,
- sur les diagonales des faces du cube,
- sur les arêtes reliant les sommets du cube.

Nous représentons (FIG.50), un exemple pour le placement de la première sphère d'une règle, dont les coordonnées sont traitées en premier par l'algorithme. Cet exemple met en exergue les arcs présents dans le calcul des forces pour établir les nouvelles coordonnées de la sphère « 2 ». Les autres sphères sont encore à leurs positions initiales.

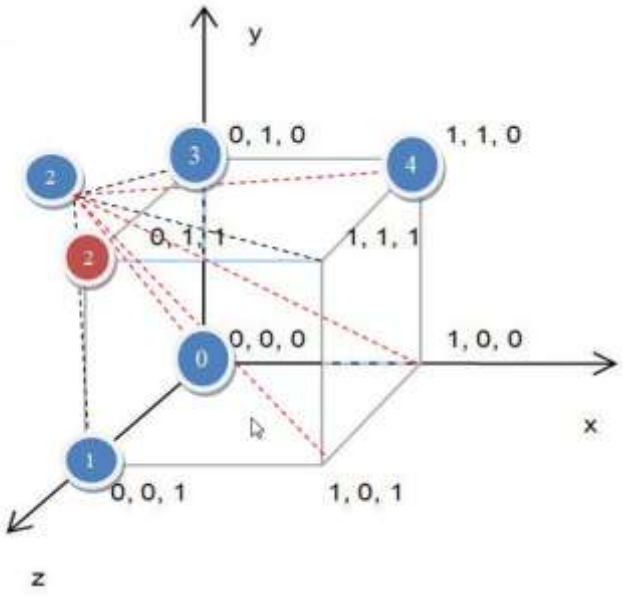


FIG.50 - Exemple de placement (sphère 2)

Si l'on suppose que le lift est identique entre chacun des attributs de la prémisse, les distances théoriques inter-sphères devraient également l'être et pouvoir interpréter de façon cohérente, les liens de corrélation graphiques. Cependant, et à partir de la construction du cube, l'obtention de distances en adéquation avec la mesure du lift semble a priori impossible. Cette impossibilité est principalement due à la prise en charge des différentes diagonales du cube pour le calcul des forces qui s'exercent entre les sphères.

Nous avons expérimenté une solution qui consiste à attribuer aux arcs de plus grande longueur, les mesures du lift les plus faibles. Le but de la métaphore étant avant tout de distinguer les liens de corrélation les plus élevés (lift élevé) par rapport aux plus faibles (lift faible). Nous cherchons par ce biais, à limiter les incohérences potentielles dans l'interprétation graphique des liens de corrélation inter-sphères.

Nous avons vu que dans une représentation 3D, l'appréciation des distances n'est pas triviale et qu'il est souvent nécessaire pour comprendre la structure d'un objet, de devoir déplacer le contrôle du point de vue ou l'objet lui-même. Nous avons donc relevé les valeurs numériques réelles des distances existantes entre les cinq sphères à l'aide de notre prototype de métaphore (TAB.16).

Le tableau met en évidence les arcs dont les longueurs sont les plus élevés par des cases grisées. Notre graphe de cinq sphères comporte une diagonale du cube identifiée ici par l'arc (6). Nous plaçons donc sur cette diagonale la valeur du lift la plus faible existante entre les attributs. Nous procédons de façon croissante suivant ce même principe pour toutes les autres valeurs du lift. Les arcs mentionnés dans le tableau forment un graphe connexe[51]. Ce graphe non planaire est représenté (FIG.51) dans une position quelconque.

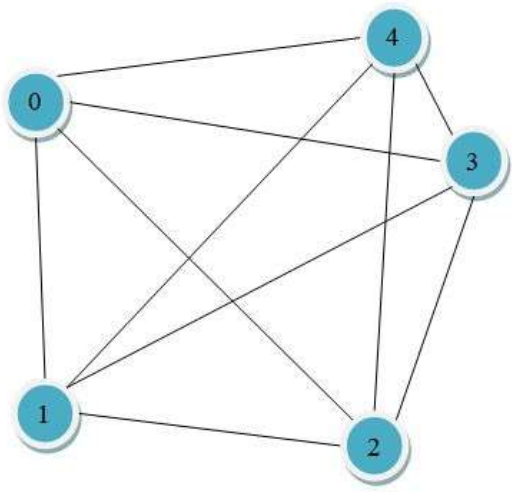


FIG.51 – Graphe de liaisons inter-sphères

id_arcs	source	destination	Lift	distances
0	0	1	**6.6**	**0.64**
1	0	2	2	0.99
2	0	3	3.3	0.83
3	0	4	1.6	1.33
4	1	2	**5**	**0.82**
5	1	3	**1.25**	**1.29**
6	1	4	**1.1**	**1.63**
7	2	3	2.5	0.97
8	2	4	1.42	1.47
9	3	4	2.8	1.24

TAB.16 - Rapport lift/distance

[51] Un graphe est dit connexe si pour toute paire de sommets distincts il existe un arc les reliant. L'orientation des arcs n'a pas d'importance pour qu'un graphe soit connexe.

Dans notre exemple, la distance la plus élevée entre les sphères se trouve donc sur l'arc [1,4] qui correspond au plus faible lien de corrélation existant entre les sphères. La distance théorique la moins élevée pour le lift le plus élevé est positionné sur l'arc [0,1] (une arête du cube). Nous constatons suivant ce principe que la valeur du lift la plus grande, correspond bien à la distance la plus petite (lift = 6.6, distance = 0.64).

Les distances inter-sphères obtenues en sortie de l'algorithme s'avèrent donc cohérentes par rapport aux valeurs respectives de la valeur du lift attribué à chacun des arcs du graphe. Le calcul des forces à l'intérieur du graphe forment un graphe complet[52] (FIG.34).

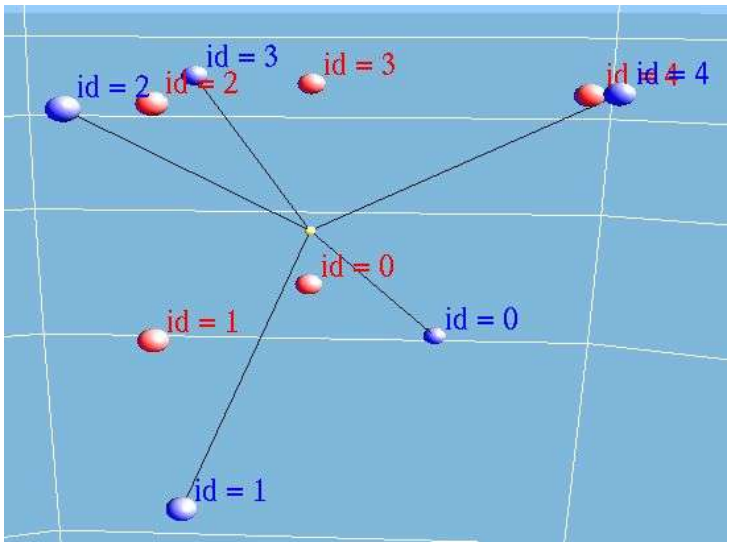


FIG.52 – Liens de corrélation graphiques dans une règle

Nous représentons (FIG.51) le graphe obtenu suivant un point de vue qui met en évidence l'éloignement des sphères 1 et 4. Notons que l'interprétation des liens de corrélation n'est pas triviale dans une représentation 3D et nécessite de déplacer le point de vue pour évaluer les distances entre les sphères sous différents « angles ». La mesure du lift est normalisée afin de contenir le graphe dans l'arène 3D, cette normalisation assure une lisibilité correcte des règles entre elles et dans l'arène 3D (FIG.53).

Cette solution fonctionne donc pour l'élaboration des liens de corrélation graphique entre les sphères. Elle nécessite néanmoins de gérer l'attribution des valeurs du lift en fonction des différences de longueur pour les arcs qui interviennent dans le calcul des forces. Les liens de corrélation graphique peuvent être interprétés. Nous avons vu que les valeurs pour la mesure du lift sont attribuées aux arcs dont les longueurs sont les pus élevées (diagonales). Les longueurs d'arc les plus faibles (liant les arêtes) seront réservés pour les valeurs du lift les plus faibles. Dans ces conditions seulement, les liens de corrélation entre les différentes sphères pourront être interprétés graphiquement.

[52] Un graphe est complet si deux sommets quelconques sont reliés dans au moins une direction.

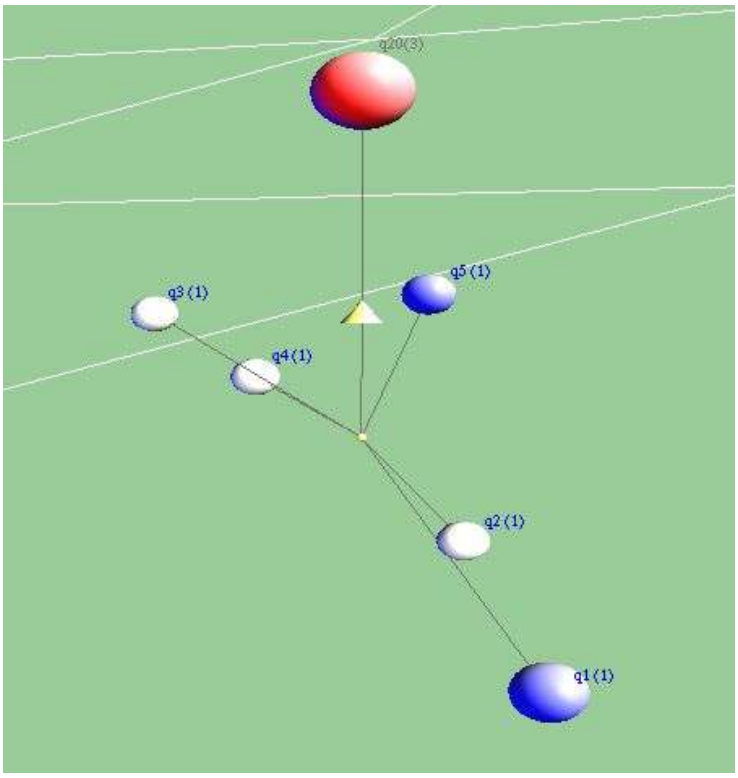


FIG.53 – Liens de corrélation graphiques dans une règle

4.7 Expérimentation sur des données réelles

4.7.1 Efficacité du système d'extraction

Les données portent sur une enquête de satisfaction effectuée par Nantes habitat auprès des locataires logés par l'Office HLM. Le volume des données réelles pour l'expérimentation de l'outil d'extraction est assez significatif pour nous confronter aux problèmes de mise au point du programme. Nous avons vu que le calcul du gain informationnel demande de nombreuses itérations. Toutes ces itérations interrogent la base de données. Il a donc fallu modifier les fonctions chargées du calcul de cette mesure, afin d'obtenir des temps de réponse acceptable du programme. Ces modifications ont demandé un effort de développement supplémentaire pour introduire les tables de cardinalités. Les premières règles apparaissent alors en quelques secondes. Cependant le parcours de l'ensemble du contexte par l'algorithme d'extraction prend encore plusieurs dizaines de minutes.

Notons que le parcours du contexte n'est pas nécessaire, puisque les règles qui comportent le plus d'intérêt au sens de l'information portée par les attributs, se trouvent en avant de la scène. Le temps de réponse du système pour extraire et afficher la centaine de règles que peut supporter l'espace représenté par l'arène est donc acceptable pour implémenter des primitives de navigation interactives (relations de voisinage voir FIG.21).

La FIG.54 montre la représentation d'une primitive d'extraction à partir de la table de données de Nantes Habitat. Bien que les mesures de support et de confiance soient très faibles pour les règles extraites, elles montrent graphiquement le gain informationnel de façon significative.

Par exemple (FIG.54), nous détaillons une règle et observons que les gains d'information apportés par *q2*, *q3* et *q4* sont négatifs (sphères de couleur blanche). Leur absence augmenterait donc la qualité de la règle. En parcourant le paysage de règles par des primitives de navigation classique (avancer, reculer, droite, gauche, etc.), nous observons que les règles décrites avec une valeur de support élevé comporte les gains d'information les plus élevés. La représentation est donc cohérente par rapport à la nature entropique de cette mesure objective.

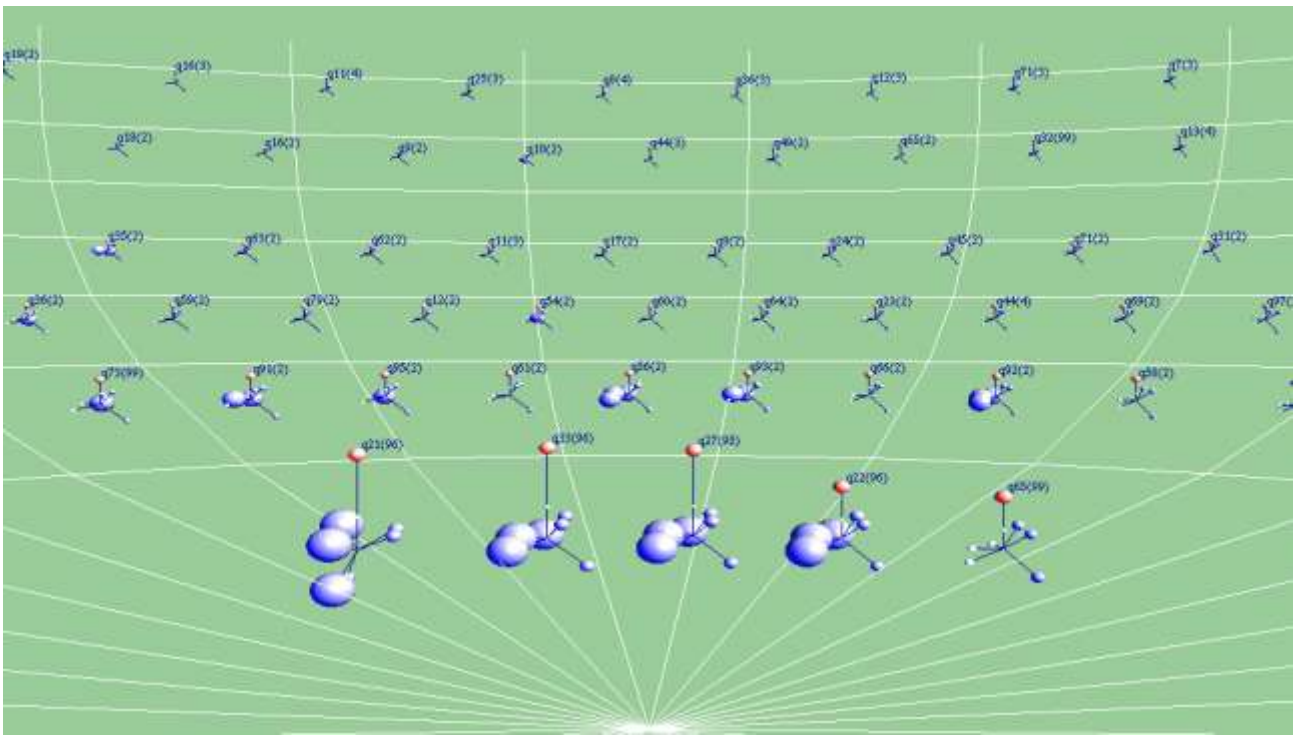


FIG.54 - Visualisation d'une primitive d'extraction (Nantes habitat)

Chapitre 5

Conclusion

Le prototype d'extraction développé dans cette étude s'appuie sur le concept de réalité virtuelle. Il présente à l'utilisateur un paysage de règles d'association dans une arène 3D. La métaphore de règles est inspirée de celle de la molécule et implémente quatre mesures d'intérêt distinctes. Elle permet une lecture graphique des deux mesures les plus utilisées, le support et la confiance, et offre la possibilité de lire graphiquement les liens de corrélation entre les attributs (lift). La quatrième mesure est une mesure objective de l'intérêt porté par chacun des attributs considérés individuellement. Nous révélons ici, l'importance de la présence ou de l'absence des attributs qui composent une règle en augmentant le niveau de granularité de l'information portée par chacun d'eux.

5.1 Acquis

Il est difficile de résumer en quelques lignes les acquis de neufs mois passés au sein d'un laboratoire d'informatique. J'ai découvert le monde de la recherche, totalement inconnu jusqu'alors. Le travail des chercheurs et des doctorants, de leurs méthodes et de leur environnement complémentaire à celui de l'entreprise. Derrière l'acronyme ECD se cachent de nombreux concepts et en particulier lorsqu'il s'agit de découvrir de la connaissance nouvelle, non triviale et non intuitive mais bien réelle. De l'évaluer ensuite par des mesures d'intérêt. Je me suis documenté pour comprendre les fondements et les techniques utilisées en ECD. Sur le plan pratique, nous avons réalisé un système client serveur pour extraire et visualiser des règles d'association à partir d'une table de données. Il m'a fallu considérer les aspects techniques (C/C++, OpenGL, SQL) et les méthodes sur lesquelles le développement d'applications est basé (UML, gestion des sources). Les considérations techniques, mathématiques et physiques pour développer ce prototype d'extraction et de visualisation de règles ont été nombreuses. Mes travaux m'ont suscité bien des interrogations avec la nécessité parfois, de « revisiter » avec bénéfices, mes connaissances tant théoriques que pratiques. Ce fut le cas notamment pour la programmation en langage C et la géométrie tridimensionnelle.

5.2 Perspectives

Nous sommes conscients avoir exploré qu'une infime partie de ce vaste domaine que représente la visualisation de l'information. En ECD, découvrir de la connaissance à l'aide d'un encodage graphique spécifique pour représenter des données brutes reste une tâche complexe. Tout d'abord parce que les variables graphiques (taille, luminosité, position, forme, etc.) doivent traduire virtuellement à partir de données réelles une représentation intellectuelle abstraites. Ce concept cache entre le réel et l'abstrait une multitude de scénarii possibles pour représenter l'information. En ECD, l'utilisation de la visualisation de l'information pour la fouille de données est un champ de recherche immense. Notre métaphore apporte une première solution en encodant quatre mesures d'intérêt, et il est certain, que de nombreuses améliorations ou modifications sont encore possibles :

- par l'encodage graphique de mesures d'intérêt supplémentaires (ex : matérialiser l'intensité d'implication par une flèche sur l'arc reliant la prémisse à la conclusion),

- par une amélioration de l'algorithme de placement et de l'évaluation des distances inter-sphères. L'élimination des perturbations engendrées par la forme géométrique initiale (cube) pour initialiser les coordonnées des solides représentant les attributs dans la scène 3D.

- l'intégration de relations de voisinages et de moyens d'interaction pour l'utilisateur, un mode d'interaction simple serait de réaliser des *slider* dont les mouvements seraient associés aux seuils des indices de qualités,

- la possibilité de sélectionner directement les attributs dans la scène soit par un menu contextuel déroulant soit par des objets graphiques 3D. Toutefois cette méthode impose une réflexion approfondie de l'ergonomie afin qu'un système offrant une distance articulatoire faible ne soit pas au prix d'une interface d'interaction trop chargée.

- par des moyens de fouilles interactives qui n'ont pas pu être implémentées dans le système par manque de temps et nécessiteraient encore de long moment de développement.

- Ce module d'extraction et de visualisation de règles est appelé par une interface en lignes de commande. Il peut donc être intégré dans des *batch* de traitements plus globaux de l'ensemble des actions de l'extraction des données initiales à la restitution visuelle des résultats. Ces *batch* seraient appelés soit par des repères dans la scène ou directement en interagissant avec les règles représentées.

- Naviguer dans les données. Dans l'outil de visualisation ARVis, les règles de voisinage spécifient ou généralisent des ensembles de règles. Spécifier ou généralisé un sous-ensemble de règles revient alors à ajouter ou respectivement à soustraire un *item* de l'*itemset*. Ces relations étudient des phénomènes de plus en plus particuliers ou de plus en plus globaux.

Enfin, nous noterons qu'en visualisation de l'information et notamment en ECD, seule une phase d'expérimentation auprès des experts métier, permet de valider l'efficacité de l'outil proposé. Le domaine de la visualisation de l'information associées à la fouille de donnés en ECD sont des axes de recherche ouverts. Les possibilités de représenter l'information par des techniques avancées n'a de limites que par l'imagination de l'homme. Les chemins restant à explorés dans ce domaine semblent alors infinis.

Liste des tableaux

TAB.1 - Règles d'association 16
TAB.2 - Mesures objectives de règles d'association [Tan et al., 2004] 20
TAB.3 - Classification des mesures d'intérêt objectives BLANCHARD et *al*., (2005) 23
TAB.4 - Extraction des *itemsets* fréquents dans *Apriori* 26
TAB.5 - Table des données 26
TAB.6 - Décomposition des sous-ensembles C_k et L_{k-1} 27
TAB.7 - Validation des règles 28
TAB.8 - Fonctionnalités *PostgreSQL* et *MySQL* 52
TAB.9 - Architecture logicielle, Java3D 57
TAB.10 - Récapitulatif des choix techniques 59
TAB.11 - Cardinalités des attributs 63
TAB.12 - Résultats des indices de qualité ($X \rightarrow Y$) 63
TAB.13 - Calcul du lift entre les attributs 63
TAB.14 - Cardinalités des ensembles 64
TAB.15 - Correspondance graphique (objets/mesures) 72
TAB.16 - Rapport lift/distance 91

Liste des figures

FIG.1 - Pilotage d'une fouille interactive de données 9
FIG.2 - L'ECD à la confluence de nombreux domaines 12
FIG.3 - Processus ECD [Fayyad et *al*., 1996] 13
FIG.4 - Panier de la ménagère. 17
FIG.5 - Table de contingence de *r* 19
FIG.6 - Trace d'exécution de CHARM 29
FIG.7 - L'explorateur de règles *IRSetNav*. 30
FIG.8 - Une matrice *item*-à-*item* de visualisation 2D et 3D dans *LARM* 32
FIG.9 - Une matrice *item*-règles 33
FIG.10 - Un graphe d'*items*(*a*), un graphe d'*itemsets*(*b*) 33
FIG.11 - Visualisation de règles par matrices (a : *DBMiner*, b : *Entreprise Miner*) et graphes (c : *DBMiner*, d : *Intelligent Miner*) BLANCHARD, (2005) 34
FIG.12 - Modèle générique pour la visualisation d'information CARD et *al*, (1999) 36
FIG.13 - Paysage d'informations, BLANCHARD (2005) 37
FIG.14 – *Fisheye*, CHEVRIN., et al (2005) 37
FIG.15 - Les variables rétiniennes de Bertin 38
FIG.16 - Conception assistée par ordinateur (*CATIA*) 40
FIG.17 - Nuages de points, sans rendu volumique (*a*), avec rendu volumique (*b*) 42
FIG.18 - Visualisation d'arbres coniques (*a*) et métaphore botanique (*b*) 42
FIG.19 - Visualisation d'un arbre dans un espace hyperbolique (*focus + context*) 43
FIG.20 - Nuages de points 3D avec TIDE 44
FIG.21 - La relation de voisinage pour naviguer parmi les sous-ensembles 45

FIG.22 - Encodage graphique dans *ARVis* 46
FIG.23 - Changement de monde dans *ARVis* 46
FIG.24 - Table du modèle de données. 48
FIG.25 - Architectures logique (a) et physique (b) 49
FIG.26 - *PgAdminIII* : outil d'administration pour *PostregSQL* 53
FIG.27 - Principe d'affichage OpenGL 55
FIG.28 - Pipe-line de rendu simplifié OpenGL 55
FIG.29 - Arborescence des objets d'un univers Java3D 57
FIG.30 - Métaphore d'une règle d'association 60
FIG.31 – Combinaison des items de la prémisse 63
FIG.32 - Loi de Coulomb, répulsion 65
FIG.33 - Effet de Hooke, attraction 66
FIG.34 - Attraction, répulsion (q_1) 67
FIG.35 - Etirement du graphe 69
FIG.36 - Coordonnées des points de concours des arêtes (repère OpenGL) 70
FIG.37 - Arène 3D (OpenGL) 71
FIG.38 - Encodage graphique des mesures d'intérêt 72
FIG.39 - Diagramme de Gantt des tâches du projet 74
FIG.40 - Architecture du processus de fouille 75
FIG.41 - Diagramme de classes simplifié 76
FIG.42 - Table de résultats (*pgAdmin*) 81
FIG.43 - Calcul des coordonnées dans l'arène 84
FIG.44 - Positionnement des règles dans l'arène 84
FIG.45 - Interface utilisateur 84
FIG.46 - Diagramme d'activité 85
FIG.47 – Diagramme d'activité (extraction des règles) 86
FIG.48 - Extraction avec *Tanagra* 87
FIG.49 - Initialisation des coordonnées 89
FIG.50 - Exemple de placement (sphère 2) 90
FIG.51 – Graphe de liaisons inter-sphères 91
FIG.52 – Liens de corrélation graphiques dans une règle 92
FIG.53 – Liens de corrélation graphiques dans une règle 93
FIG.54 - Visualisation d'une primitive d'extraction (Nantes habitat) 95

Bibliographie

ABISROR J.M., 2002. *VRML97 sous Java-OpenGL*. Mémoire d'Ingénieur CNAM, Juin.
url : http://cedric.cnam.fr/PUBLIS/RC396.pdf

ALESSANDRI M., 2010. *Extraction de motifs fréquents pertinents pour la préconisation d'achats en marketing one-to-one*. Mémoire d'Ingénieur CNAM, Mars.

AGGARWAL C. C., 2002. *Towards effective and interpretable data mining by visual interaction SIGKDD Explorations*, **3**, 11-22.

AGRAWAL R., IMIELINSKI T., SWAMI A., 1993. *Mining association rules between sets of items in large databases*. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, Washington D.C., 207–216.
url: http://rakesh.agrawal-family.com/papers/sigmod93assoc.pdf.

BEN-SAID Z., GUILLET F., RICHARD P., 2010. *Classification des techniques de fouille visuelle de données en 3d et réalité virtuelle*. In *EGC*.

BEN SAID Z., GUILLET F., RICHARD P., 2010. *3D visualization and virtual reality for visual data mining* : *A survey*. In *International Conference on Information Visualization Theory and Applications* (IVAPP'10), 2010.

BLANCHARD J., 2005. *Un système de visualisation pour l'extraction, l'évaluation, et l'exploration interactives des règles d'association*. Thèse de docteur es-Informatique, Ecole des mines de Nantes, 183 p.
url: http://www.polytech.univ-nantes.fr/blanchard/These.pdf

BLANCHARD J., GUILLET F., BRIAND F., GRAS R., 2005. *Assessing rule interestingness with a probabilistic measure of deviation from equilibrium*. In *Proceedings of the 11th international symposium on Applied Stochastic Models and Data Analysis ASMDA*, 191–200.
url: http://conferences.telecom-bretagne.eu/asmda2005/IMG/pdf/proceedings/191.pdf.

BERTIN J., *Sémiologie graphique. Les diagrammes. Les réseaux. Les cartes, Archives des sciences sociales des religions*, 1968, vol. 26, n° 1, p. 176-177.

CARD S. K., MACKINLAY J., SHEINEIDERMAN B., 1999. *Readings in information visualization: Using vision to think*. Morgan Kaufmann.

CHEN C. 2004. *Information visualization: beyond the horizon*.

CHEVRIN V., COUTURIER O., MEPHU NGUIFO E., ROUILLARD J., 2007. *Recherche anthropocentrée de règles d'association pour l'aide à la décision*. Université d'Artois, CRIL IUT de Lens, France.

COUTURIER O., 2005. *Contribution à la fouille de données : règles d'association et interactivité au sein d'un processus d'extraction de connaissances dans les données*. Thèse d'Université, Université d'Artois, CRIL, Lens, France, Décembre.

FAYYAD U.M. GRINSTEIN G.G., WIERSE A., 2001. *Information visualization in data mining and knowledge discovery*. San Francisco: Morgan Kaufmann publishers

FAYYAD U., PIATETSKY-SHAPIRO G., SMYTH P. *From Data Mining to Knowledge Discovery*, 1996: an overview. *AI Mag*., **17**, 1-34.
url: http://www.aaai.org/ojs/index.php/aimagazine/article/view/1230/1131

FRAWLEY W. J., PIATESKY-SHAPIRO G., MATHEUS C. J., 1992. *Knowledge discovery in databases: an overview*. *AI Mag*., **13**, 57-70.
url: http://www.aaai.org/ojs/index.php/aimagazine/article/view/1011/929

FREITAS A.A. 1998. *On objective measures of rule surprisingness*. Lecture Notes In *Artificial Intelligence 1510: Principles of Data Mining and KnowledgeDiscovery (Proc. 2nd European Symp., PKDD´98, Nantes, France)*, 1-9.Springer-Verlag.
url: http://citeseerx.ist.psu.edu

FULE P., RODDICK J.F., 2004. *Experiences in building a tool for navigating association rule result sets*. In *CRPIT'04: Proceedings of the second Australasian workshop on information security, data mining, web intelligence, and software internationalization* (HOGAN J., MONTAGE P., PURVIS M., STEKETEE C.), Eds, Computer Society, p. 103-108.

FUCHS P., MOREAU G., PAPIN J., 2001. *Le traité de la réalité virtuelle*. Les presses de l'école des Mines de Paris.

TONIC F., (2010). *Base de données, choisir et optimiser*, *Revue Programmez*, Janvier, 16-20

IMIELINSKI T., MANNILA H., 1996. *A database perspective on knowledge discovery. Communications of the ACM*. **39**, 58-64.

JOHNSON A., LEIGH J., *Tele-immersive collaboration in the CAVE research* network. In *Collaborative Virtual Environments digital places and spaces for interaction* (E. Churchill, D. Snowdon & A. Munro, Eds. Springer-Verlag, 2001, p. 225–243.

KLEIBERG E., VAN DE WETERING H., WIJK J. J. V., 2001. *Botanical visualization of huge hierarchies*. In *INFOVIS'01: Proceedings of the IEEE Symposium on Information Visualization (INFOVIS'01)*, *IEEE Computer Society*, 87–94.

KUNTZ P., LEHN R., GUILLET F., PINAUD B., 2006. Découverte interactive de règles d'association via une interface visuelle. *Visualisation en Extraction des Connaissances*. Eds, Cépaduès, LINA-COD, 113-125.
url: http://hal.archives-ouvertes.fr/docs/00/33/59/51/PDF/visu-RNTI2611.pdf

LEHN R., 2000. Un système interactif de visualisation et de fouille de règles pour l'extraction connaissances dans les bases de données. Thèse de docteur es-Informatique. Université de Nantes.

LIU B., HSY W., CHEN S., 1997. *Using general impressions to analyze discovered classification rules*. In *Proceedings of the 3rd international conference on knowledge discovery and data mining.*
url: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.51.8236&rep=rep1&type=pdf.

LIU B., HSY W., MA Y. *Pruning and summarizing the discovered associations*. In *proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining* (KDD-99), 125–134.
url: http://www.cs.uic.edu/~liub/publications/papers_topics.html

MA Y., LIU B., WRONG C.K., 2000. *Web for data mining: organizing and interpreting the discovered rules using the web. SIGKDD Explorations*, **2**, 16-23.

MUNZNER T., 2000. Interactive visualization of large graphs and networks, PhD thesis, Stanford University.

PIATETSKY-SHAPIRO G., 1991. *Discovery, Analysis, and Presentation of Strong Rules*. *AAAI/MIT*. Press, ISBN 0-262-62080-4.

RAKOTOMALALA R., 2005. Tanagra : un logiciel gratuit pour l'enseignement et la recherche. In Actes de EGC 2005, RNTI-E-3, **2**, 697-702.
url: http://eric.univ-lyon2.fr/~ricco/tanagra/fr/tanagra.html.

TAN P.N., KUMAR V., SRIVASTAVA J., 2002. *Selecting the right objective measure for association patterns. Actes ACM SIGKDD*. In *international conference of knowledge discovery and data mining*. Edmonton, Canada.

TAN P.N., KUMAR V., SRIVASTAVA J., 2004. *Selecting the right objective measure for association analysis*. Information Systems, **29**, 293–313.
url: http://www.cse.msu.edu/~ptan/papers/IS.pdf.

WEGMAN J., SYMANZIK J., 2002. *Immersive projection technology for visual data mining.* Journal of Computational and Graphical Statistics **1**, 163-188.

ZIGHED D.A., RAKOTOMALA R., 2002. Extraction de connaissances à partir de données (ECD). *Technique de l'ingénieur,* **H3744** 1-24.

ZAHER, M. H. Programmer en C++ avec PostgreSQL & Visual C++. *Association Tunisienne des Logiciels Libres*, **2002**.

ZAKI M. J., HSIAO C. J., 2002. *CHARM: An efficient algorithm for closed itemset mining.* In *Proceedings of the Second SIAM International Conference on Data Mining*. Arlington, 1-17.
url: http://making.csie.ndhu.edu.tw/course/2008/Spring/Data_Mining/paper/sdm02-27.pdf.

Lexique

A

Aprori

- algorithme d'extraction de règles d'associations [Agrawal and Srikant, 1994].

ARVis

- acronyme pour *Association Rules visualization* : logiciel de visualisation de règles d'association dans une scène 3D basée sur la technologie VRML.

API

- acronyme pour "*Application Programming Interface*".

B

BdD

- acronyme pour "Base de données ».

C

CATIA

- acronyme pour **«** Conception Assistée Tridimensionnelle Interactive Appliquée »

Confiance

- pourcentage d'itemsets contenant la prémisse d'une règle qui contiennent aussi la conclusion.

CSV

- acronyme pour "*Comma-Separated Values*". Format simple de stockage de données sous la forme d'articles et de champs délimités par des séparateurs.

E

ECD

- acronyme pour « *Extraction de Connaissances dans les Données* ». Également appelée data-mining" ou "fouille de données". "Extraction non triviale de connaissances implicites, inconnues au préalable et potentiellement intéressantes contenues dans des données" [Frawley et al., 1992]. Domaine de connaissance couvrant les différentes techniques de recherche de connaissances par des moyens informatiques dans de grands volumes de données.

I

Item

- valeur (ou modalité) particulière d'un attribut.

Itemset

- ensemble d'items.

Itemset fréquent

- itemset dont le support est supérieur au support minimal défini au préalable.

L

Lift

- mesure liée à la dépendance des liens existants entre la prémisse et la conclusion d'une règle d'association.

O

OpenGL

- acronyme pour « Open Graphics Library » c'est une spécification qui définit une API multiplate-forme pour la conception d'applications générant des images 3D.

P

PMML

- acronyme pour "*Predictive Model Markup Language*". Format d'échange de données en data-mining basé sur la syntaxe XML.

S

Support

- pour une règle : nombre de transactions qui vérifient la règle. Pour un itemset : nombre de transactions comportant tous ses items.

R

RV

- réalité virtuelle : simulation informatique interactive, immersive, visuelle, sonore et/ou haptique, d'environnements réels ou imaginaires.

S

SGBD

- Système de gestion de bases de données.

U

UML

- acronyme pour "*Unified Modeling Langage*". Norme de définition des données et des traitements sous forme de graphes suivant une méthodologie objet. Facilite la conception d'applications par des méthodes graphiques rendant la communication entre les différents acteurs du projet plus aisée et par l'intégration des concepts objet (héritage, polymorphisme).

V

VRML

- acronyme pour « *Virtual Reality Modeling Language* ». C'est st un langage de description d'univers virtuels en trois dimensions.

Résumé

Ce mémoire s'intéresse à la visualisation de règles d'association dans une représentation tridimensionnelle. Nous abordons une problématique en fouille de données qui consiste à faire face aux gros volumes de règles extraites par les algorithmes classiques. Pour ce faire, nous cherchons à insérer l'utilisateur dans la boucle de recherche de la connaissance par une représentation graphique interactive. La visualisation de l'information diminue l'effort cognitif de l'utilisateur pour évaluer et valider les règles. En ECD, la difficulté de la fouille visuelle de données est alors de traduire les données brutes abstraites sous formes graphiques concrètes, compréhensibles et porteuses de sens. Ce concept relève de la métaphore. La partie bibliographique décrit l'outillage théorique et technique disponible : les algorithmes d'extraction et les différentes méthodes de visualisation de l'information empruntées aux techniques de réalité virtuelle.
La partie réalisation décrit l'outil d'extraction qui interroge une base de données *PostgreSQL*. La partie sur la visualisation 3D présente la nouvelle métaphore de règles et les mesures qui la définissent. Nous implémentons une mesure objective FREITAS A .A, (1998) qui montre l'importance de l'interaction entre les attributs. Cette notion cruciale dans la recherche de la connaissance n'est pas très répandue dans la littérature. Dans notre métaphore, elle offre une lecture directe des différents types de règles qui portent de l'intérêt, en mesurant le gain d'information apporté par chacun des attributs qu'elle comporte.

Mots clé : ECD, règles d'association, métaphore

Abstract

The purpose of this report is to provide a three-dimensional graphic representation solution for association rules. The principal disadvantage in the process of rule generation from databases is the huge volumes of rules extracted by classic algorithms. Our approach aims at integrating the human in the data exploration process. To do it, we try to insert the user like a decision-maker into a search loop for the knowledge discovering by an interactive graphic representation. The information visualization greatly reduces the cognitive effort for user in charge of estimate and validation rules extracted. In ECD, the difficulty of the visual data mining is then to translate raw abstract data into concrete shapes graphic. This concept is based on metaphor. The related works part presents the available tools and techniques: extract association rules algorithms and methods to display information on screen or using virtual reality.
The realization part describes the extraction tool using a *PostgreSQL* database. The visualization part presents a new metaphor for rules description and measures. We implement an objective measure FREITAS A .A, (1998) which shows the importance of the interaction between the attributes belonging to the rule antecedent. This key concept of data mining has been relatively little investigated in the literature. Our metaphor offers a direct reading of the various kinds of rules who carry interest by measuring the information gain through each individual attributes.

Keywords: data-mining, association rules, metaphor