# Bases de données multidimensionnelles floues et leur utilisation pour la fouille de données

## THÈSE DE DOCTORAT DE L'UNIVERSITÉ PARIS 6

présentée pour obtenir le grade de

**DOCTEUR DE L'UNIVERSITE PARIS 6**

**(spécialité informatique)**

par

Anne Laurent

**Soutenue le 20 septembre 2002 devant le jury composé de**

P. BOSC (Président)
B. BOUCHON-MEUNIER (Directrice de thèse)
J-F BOULICAUT (Rapporteur)
F. DENIS (Examinateur)
E. DIDAY (Examinateur)
A. DOUCET (Co-directrice)
H.L. LARSEN (Rapporteur)
C. MARSALA (Examinateur)
C. TIJUS (Co-directeur)

# Résumé

Devant l'accroissement des volumes de données à traiter, les entrepôts de données se sont développés depuis le début des années 1990 afin de fournir aux décideurs des systèmes dédiés à l'analyse des données. Les systèmes opérationnels, utilisés pour les applications transactionnelles (OLTP - *On-Line Transaction Processing*), se sont vite révélés inadaptés pour les environnements décisionnels. Les bases de données multidimensionnelles ont donc émergé pour répondre aux besoins spécifiques d'analyse multidimensionnelle en ligne (OLAP - *On-Line Analytical Processing*). Toutefois, les données issues du monde réel sont souvent entachées d'imperfection et les requêtes que les utilisateurs souhaitent poser ont souvent des formulations vagues. La théorie des sous-ensembles flous permet d'améliorer les systèmes d'information, notamment pour l'interprétation qualitative de données quantitatives (numériques). Les informations présentées à l'utilisateur (sous forme de règles par exemple) sont plus compréhensibles et se généralisent mieux. L'objectif de notre travail est de définir un cadre formel pour la mise en œuvre de systèmes de fouille de données floues avec des outils OLAP. Nous proposons une extension du modèle multidimensionnel pour le traitement de données imparfaitement connues et de requêtes flexibles. Ce modèle étendu est intégré dans une architecture comprenant des outils de fouille de données floue. En particulier, nous introduisons un système pour la construction d'arbres de décision flous et la génération automatique de résumés linguistiques à partir de bases de données multidimensionnelles floues, utilisées comme support de fouille de données. Nous étudions également le problème des cellules vides, sous ses aspects sémantiques, pour la recherche d'anomalies.



**Mots-clés:** bases de données multidimensionnelles, OLAP, fouille de données, logique floue, résumés flous.

# Abstract

Since the early 1990s, data warehousing has provided an efficient framework to deal with huge amounts of data in an analytical view. It has appeared that transactional systems (OLTP - *On-Line Transactional Processing*) are not suitable for fast multidimensional analysis. For this reason, multidimensional databases have emerged to support OLAP tools (*On-Line Analytical Processing*). However, real world data are often imperfect, being either uncertain, or imprecise. This explains why it is important for database management systems to be able to cope with flexible queries. In this framework, fuzzy set theory is appropriate for improving information systems in particular for qualitative interpretation of quantitative data (numerical data). The information provided to users (for instance by means of rules) is more understandable and generalizable. Our work aims at defining an approach to perform OLAP-based mining using fuzzy multidimensional databases and fuzzy data mining algorithms. We propose an extension of multidimensional databases to handle imperfect information and flexible multidimensional queries. We also integrate fuzzy multidimensional databases with machine learning methods. More specifically, using the proposed model, we introduce a system that allows to build fuzzy decision trees and to generate linguistic summaries using the proposed data model. A general architecture is provided, based on fuzzy multidimensional databases as a support for knowledge discovery. Finally, we study from a semantic point of view the problem of abnomalies related to the presence of empty cells.



**Keywords:** Multidimensional Databases, OLAP, Data Mining, Fuzzy Logic, Fuzzy Summaries.

# Remerciements

Je tiens tout d'abord à remercier le plus chaleureusement possible Bernadette Bouchon-Meunier. Durant ces trois années de thèse, et avant lors de mon stage de DEA, elle m'a transmis le goût du travail de recherche. D'une grande disponibilité, Bernadette Bouchon-Meunier m'a laissé une grande liberté sur mon travail, et m'a éclairée de ses nombreux excellents conseils, suggestions et remarques. C'est avec le plus grand plaisir que j'ai travaillé dans son équipe, et la séparation de cette fin de thèse n'en est que plus difficile, en espérant simplement que nos chemins ne se séparent pas tout à fait.

Je tiens également à exprimer mes remerciements à Anne Doucet, pour avoir accepté de co-diriger ma thèse. Anne Doucet a orienté mon travail de recherche grâce aux discussions fructueuses que nous avons eues et aux relectures attentives des nombreux documents que je lui ai soumis tout au long de ces trois années.

Mon travail avec Charles Tijus a été très enrichissant, et je le remercie vivement d'avoir accepté de co-diriger mon travail. Le point de vue cognitif constitue un apport très intéressant dans la démarche adoptée dans ma thèse, et Charles Tijus m'a apporté un éclairage nouveau, même si je regrette de n'avoir pas pu consacrer plus de temps à cette démarche.

Je tiens à remercier Jean-François Boulicaut pour avoir accepté d'être rapporteur de ma thèse. Ses compétences scientifiques et son ouverture d'esprit sur la logique floue en font le rapporteur idéal. Les discussions que nous avons eues tout au long de ces trois années où nous avons pu nous croiser lors de conférences ont contribué à l'amélioration de ce mémoire.

Je tiens également à remercier H.L. Larsen pour avoir aussi accepté d'être rapporteur de cette thèse. La tâche n'était pas aisée puisque le français n'est pas sa langue et je le remercie donc d'avoir pris de son temps précieux. Ses remarques ont contribué à l'amélioration du mémoire et ont apporté de nouvelles orientations à ma réflexion, notamment en ce qui concerne les requêtes complexes.

Christophe Marsala tient une place toute particulière dans ce travail, puisqu'il m'a encadrée tout au long de mon stage de DEA et a suivi de près mon travail de thèse. Pour tout le temps qu'il a consacré à la relecture de mes articles, pour sa grande disponibilité, sa gentillesse, et la pertinence de ses suggestions, je tiens à lui exprimer ma profonde reconnaissance.

Je tiens également à remercier Patrick Bosc, pour avoir accepté de faire partie de mon jury. Ses remarques et suggestions au cours de ma thèse et sur le manuscript ont été très bénéfiques à ma réflexion et à mon travail, et ont largement contribué à l'amélioration du mémoire.

Je tiens à remercier François Denis d'avoir accepté de participer à mon jury. Son expertise en apprentissage automatique et ses remarques ouvrent de nombreuses perspectives à mon travail. Je le remercie également pour l'aide qu'il m'a apportée pour intégrer l'université de Provence à Marseille.

Enfin, je tiens à remercier vivement Edwin Diday pour son accompagnement tout au long de ma thèse, pour ses encouragements, pour son invitation au séminaire de l'Université Paris Dauphine, et pour avoir accepté de participer à mon jury.

Ce travail n'aurait sans doute jamais abouti sans le soutien technique et amical de Jérôme Poirier, alors administrateur bases de données du DESS GLA, qui m'a beaucoup aidée à dompter Oracle.

Je remercie vivement le Ministère de l'Education Nationale, de la Recherche et de la Technologie, et en particulier Jean-Paul Dispagne, pour nous avoir fourni la base de données sur laquelle s'appuie cette thèse, et pour l'attention qu'il a portée à ce travail.

Je tiens également à remercier les membres du Laboratoire d'Informatique de Paris 6 (LIP6) où j'ai effectué cette thèse, avec une mention particulière à Ghislaine Mary et Christophe Bouder pour leur grande disponibilité, leur gentillesse et leur aide précieuse.

Je voudrais également remercier l'équipe pédagogique de l'institut Galilée de l'université Paris 13, où j'ai été monitrice pendant ces trois années de thèse. En particulier, je remercie Philippe Dague, Daniel Kayser, Adeline Nazarenko, et Jaqueline Vauzeilles qui m'ont donné le goût d'enseigner et de travailler en équipe.

Je n'oublie bien sûr pas mes amis, qui ont toujours été présents, qui ont suivi mon travail, et m'ont soutenue.

Il y a d'abord les *anciennes*, les thésardes qui finissaient quand j'ai commencé, Anne, Claudia, Ilham, Nara et Nathalie, qui m'ont communiqué leurs précieux conseils et ont rendu très agréable mon arrivée au sein du LIP6.

De l'équipe LOFTI, je garderai le meilleur souvenir et j'en remercie tous les membres, notamment Florence et Maria.

Mais de cette équipe, je tiens à adresser des remerciements tout particuliers à la sous-équipe du bureau C765, pour les formidables moments partagés, les "sorties piscine", les discussions animées, et les débats politiques, scientifiques, littéraires et culturels qui faisaient des premiers instants de la matinée un moment d'ouverture privilégié sur le monde. Merci donc à Laure, Liva, et Marie-Jeanne.

Il y a aussi les ex-collègues rencontrés lors de mes incursions dans le monde du privé, et qui sont restés des amis précieux. En particulier, que soient ici remerciés le plus chaleureusement possible Marc et Samia (et sa famille). Ce travail n'aurait sans doute jamais été le même sans les magnifiques moments passés avec eux.

Je tiens également à remercier les amis de la *promo 99* du magistère, et notamment Cédric, Frédéric, Guillaume, Mélanie, Nicolas et Romain.

Enfin, je tiens à remercier ma famille, notamment mes parents, ma soeur et Géraldine, ainsi que, *last but not least,* Patrice.

# Table des matières

# Introduction

Il est couramment estimé que le volume de données disponibles double tous les dix-huit mois [BOA 01]. Cette tendance est encore plus marquée dans certains domaines comme l'internet ou la bio-informatique [ROB 00]. La gestion efficace de ces données devient donc un problème crucial. Apparus au début des années 1990, les entrepôts de données visent à pallier ces problèmes par la mise en place de systèmes dédiés à l'intégration de données volumineuses issues de sources hétérogènes (provenant de systèmes transactionnels principalement) à des fins d'analyse. Résolument orientés vers les décideurs, utilisateurs finaux de ces environnements, les études sur ces systèmes ont suggéré de nombreuses problématiques de recherche, des fondements théoriques étant absolument nécessaires afin d'assurer la robustesse des solutions proposées. Parmi ces problématiques se trouvent notamment la gestion efficace des données, l'évaluation de leur qualité, et la mise en œuvre d'outils d'analyse et d'aide à la décision. La construction des entrepôts suppose les opérations suivantes : extraction, nettoyage, transformation, chargement, mises à jour.

La figure 1 détaille une architecture classiquement rencontrée. Les magasins de données, construits à partir des entrepôts afin de réduire le volume des données à traiter par la mise en place de *vues métier*, sont utilisés comme support d'analyse au travers de différents outils de fouille de données ou de production de rapports afin de fournir à l'utilisateur une connaissance pertinente et utile.

Dans notre étude, nous nous intéressons particulièrement aux magasins *multidimensionnels*, c'est-à-dire aux magasins qui sont des bases de données multidimensionnelles. Cette nouvelle forme de représentation des données a récemment émergé et sous-tend les besoins croissants en analyse OLAP (On-Line Analytical Processing). Ce modèle présente en effet des avantages qui sont détaillés dans la suite de cette thèse. Parmi ceux-ci, on retiendra son intérêt, d'une part dans l'optique de la fouille de données (performance des calculs d'agrégats, gestion des hiérarchies ...), et d'autre part pour la visualisation des données. Les magasins multidimensionnels constituent les bases d'apprentissage, sur lesquels différents algorithmes peuvent être mis en œuvre. De cet apprentissage sont issues des connaissances présentées à l'utilisateur.

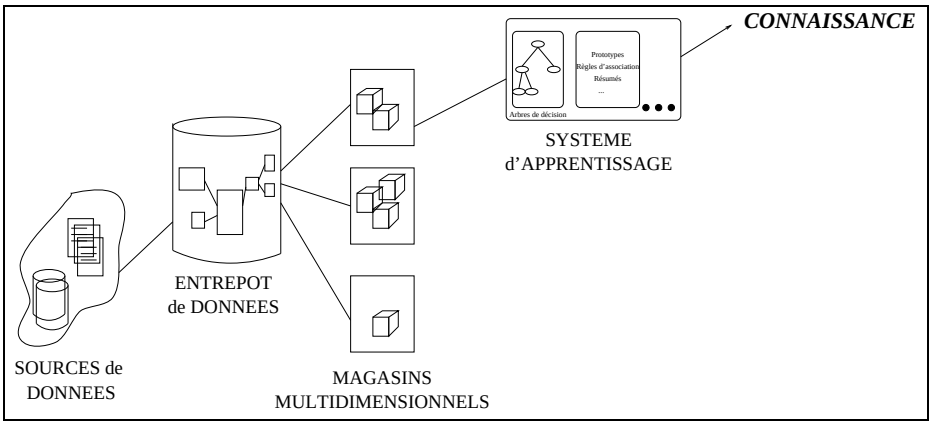


FIG. 1 – Processus général de fouille de données

Dans notre approche, ce processus de fouille de données est étendu pour la gestion des données issues du monde réel, dès la construction des bases multidimensionnelles. En effet, ces données sont souvent entachées d'imprécisions et d'incertitudes. Or les outils existants ne permettent pas la gestion de ces imperfections, ni au niveau de la représentation, ni au

niveau de l'apprentissage à partir de bases multidimensionnelles. De plus, les requêtes que les analystes souhaitent pouvoir mettre en œuvre peuvent être formulées d'une manière vague. Par exemple, on envisage souvent des requêtes de sélection des ventes *faibles* plutôt que des requêtes de sélection des ventes inférieures à un seuil strict (*12000 unités* par exemple).

Les algorithmes de fouille de données mis en œuvre ici sont étudiés pour la prise en compte de la notion d'imprécision. Ainsi, dans le cas de la construction d'arbres de décision flous par exemple, ils permettent la construction de partitions floues des domaines d'attributs numériques. Les arbres construits grâce à l'utilisation de la théorie des sous-ensembles flous sont plus robustes lors de la phase de généralisation (notamment parce que les seuils de passage d'un concept à l'autre sont plus souples), plus concis et autorisent une représentation en termes naturels des attributs numériques.

Nous nous focalisons ici sur la partie du processus allant de la construction des cubes à la présentation des connaissances produites. Dans tout ce processus, les données considérées sont potentiellement imparfaites (imprécises et/ou incertaines). Cependant toutes les solutions présentées s'appliquent également aux données *classiques* (précises et certaines).

Dans l'approche que nous proposons, nous étudions l'intégration du flou dans le processus de fouille de données, tant au niveau des bases multidimensionnelles (représentation et manipulation des connaissances), qu'au niveau de leur intégration dans le processus global de fouille de données. En particulier, nous considérons le couplage d'algorithmes de construction d'arbres de décision flous et de génération de résumés flous avec une base de données multidimensionnelles floues. En outre, nous étudions le problème posé par la découverte de cellules anormalement vides dans les cubes de données afin de déceler d'éventuelles exceptions.

Ainsi, nos travaux visent non seulement à représenter et manipuler les données classiques et imparfaites, mais également à mettre en œuvre des solutions de fouille de données afin d'extraire des connaissances pertinentes sur les données représentées de manière multidimensionnelle. Cette approche est novatrice, aucun système intégrant les données floues et l'apprentissage flou dans le processus de fouille de données à partir de bases de données multidimensionnelles n'existant à notre connaissance.

La thèse est organisée en quatre parties. La **première partie** expose l'état de l'art sur les bases de données statistiques (chapitre I.1), les bases de données multidimensionnelles (chapitre I.2), et les bases de données floues (chapitre I.3). Les trois parties suivantes sont consacrées à la présentation de nos propositions.

La **deuxième partie** est consacrée au modèle de base de données multidimensionnelles floues que nous proposons. La représentation des connaissances est détaillée dans le chapitre II.1, et la manipulation des données multidimensionnelles floues dans les chapitres II.2 et II.3. Les requêtes et leurs combinaisons sont étudiées dans le chapitre II.4.

La **troisième partie** expose les moyens mis en œuvre pour utiliser notre modèle de bases de données multidimensionnelles dans le cadre du processus de fouille de données. Le chapitre III.1 présente les systèmes de la littérature et introduit notre modèle. Les chapitres suivants sont consacrés à la présentation des méthodes de fouille de données étudiées : construction d'arbres de décision (chapitre III.2), génération de résumés flous (chapitre III.3), et découverte des cellules anormalement vides (chapitre III.4).

La **quatrième partie** présente l'implantation logicielle de notre approche et les résultats obtenus pour la fouille de données. Le chapitre IV.1 présente le système *FUB*. Le

chapitre IV.2 présente l'architecture globale de *FUB Miner*. Le chapitre IV.3 est dédié à la construction d'arbres de décision flous, le chapitre IV.4 à la génération de résumés flous et le chapitre IV.5 à la détection de cellules anormalement vides.

Enfin, nous concluons en rappelant nos contributions ainsi que les perspectives associées.

# Première partie

# Bases de données décisionnelles

Les bases de données se sont développées à partir des années 60, d'abord sur des modèles de bases réseaux et de bases hiérarchiques, puis sur le modèle relationnel dans les années 70. Les bases de données décisionnelles, tournées vers les analystes et non plus les utilisateurs de bases opérationnelles, sont aujourd'hui un thème important de recherche. Ce type de bases de données est de plus en plus utilisé, notamment par le biais des bases de données multidimensionnelles, dont l'essor est principalement dû à leur utilisation massive par les décideurs des entreprises.

Placés au cœur des systèmes d'information, les systèmes de gestion de bases de données jouent un rôle de plus en plus important dans les applications décisionnelles. Face à l'explosion du volume de données disponibles, les concepteurs de systèmes d'analyse (statistique, apprentissage notamment) se tournent vers les systèmes de gestion de bases de données pour pallier les problèmes d'accès efficace aux données.

Étudié en partie dans le domaine des bases de données statistiques, le contexte décisionnel offre un large éventail de problèmes tant théoriques que pratiques. Dans cette partie, nous étudions donc les principaux modèles de bases de données utilisés dans les systèmes dédiés aux décideurs pour l'analyse de données.

Le chapitre 1 présente brièvement les principales caractéristiques des bases de données statistiques, les bases de données multidimensionnelles faisant l'objet du chapitre 2. Enfin, le chapitre 3 introduit les principales recherches concernant les bases de données floues.

# Chapitre 1

# Bases de données statistiques

A la fin des années 1970, il est apparu que le volume des données scientifiques collectées et leur nature même nécessitaient des capacités de stockage et d'analyse qui dépassaient le savoir-faire des systèmes de bases de données existants. Sous l'impulsion à la fois des statisticiens et des informaticiens, une nouvelle voie de recherche s'est donc créée pour faciliter l'analyse de grosses bases de données, et les bases de données statistiques sont apparues entre la fin des années 1970 et le tout début des années 1980 [THO 83]. Souvent décrites comme étant des *bases de données fournissant des informations statistiques aux utilisateurs*, elles permettent de collecter des données à des fins d'analyse, et contiennent à la fois des paramètres et des variables à étudier. L'analyse provient de traitements statistiques simples (somme, moyenne par exemple) ou complexes (régression multivariée par exemple).

## 1.1 Structures de données

Les principales caractéristiques des bases statistiques visent à fournir aux utilisateurs des outils d'analyse des données scientifiques ou socio-économiques par exemple. Ces données, très présentes dans les secteurs scientifiques, gouvernementaux ou encore industriels, requièrent des systèmes spécifiques [SHO 82, CHA 83, DEN 83, OZS 83]. Dépassés par les volumes de données engendrés dans ces secteurs, les statisticiens ont ressenti le besoin d' utiliser des systèmes de gestion efficaces de ces données. Les problèmes liés à l'utilisation de telles bases sont divers. En effet, les données ne sont pas du même type que dans les systèmes opérationnels existant jusqu'alors, notamment parce qu'il s'agit de les représenter et de les traiter soit de manière résumée, par groupes d'individus, ou par sous-ensembles. De plus, des outils d'analyse statistique puissants sont requis. Pour toutes ces raisons, les systèmes relationnels *classiques* se révèlent alors inadaptés.

### 1.1.1 Types des données traitées

Les données sont distinguées selon qu'elles sont des valeurs de paramètres (*category attributes*) ou de variables (*summary attributes*), appelées également mesures. Par exemple dans le domaine scientifique, on distinguera les paramètres d'une expérience et les résultats mesurés en fonction de ces paramètres.

Les ensembles résumés (mesures/variables) sont construits par agrégation des données de la base. Les opérations d'agrégation et de désagrégation sont possibles, afin de repré-

senter les informations à différents niveaux de granularité. Des hiérarchies de classification organisées en graphes représentent les paramètres le long de ces différents niveaux d'agrégation.

Deux types de modélisation des données sont utilisées, soit sous forme de tables, soit sous forme de graphes.

Les méta-données sont beaucoup plus complexes dans le cadre des bases de données statistiques que dans le cadre relationnel. Ainsi, des informations sont stockées concernant les données manquantes, la qualité des données, l'historique de création et modification de la base, etc. Toutefois, la distinction entre données et méta-données n'est pas toujours évidente, par exemple en ce qui concerne les informations sur les attributs catégories.

En 1990, [RAF 90] a étudié les limites des systèmes de bases de données statistiques présentés jusqu'alors, et a proposé un nouveau modèle. Les auteurs ont de plus exploré les conditions nécessaires pour qu'un objet statistique soit *bien formé* et les conditions pour qu'il puisse être résumé à l'aide d'une fonction statistique.

### 1.1.2 Gestion des données

Les systèmes de gestion de bases de données existant avant l'émergence des bases statistiques ont été très peu utilisés à des fins d'analyse de données. [BRA 81] avance six raisons à cela, dont notamment le coût élevé des systèmes de gestion de bases de données conventionnels à l'époque, le fait que ces systèmes n'étaient pas assez aisés à manipuler pour des utilisateurs non experts, ou encore le manque de fonctionnalités dédiées aux analyses statistiques.

Cependant, si dans les premières années de recherche d'autres systèmes ont été développés, dans [GHO 88, GHO 91], les auteurs proposent plutôt d'étendre le modèle relationnel et l'algèbre relationnelle au contexte statistique. Ces travaux décrivent une *algèbre relationnelle statistique*, fournissant les outils statistiques nécessaires sans altérer les propriétés algébriques des langages existants. [GHO 88] propose notamment les extensions nécessaires à la création de tables résumés décrivant la distribution multivariée des données. Comme le souligne l'auteur, il est très difficile de savoir si le système est complet et couvre toutes les fonctionnalités statistiques qu'un utilisateur pourrait souhaiter.

Le volume des données à stocker a motivé de nombreux travaux sur la compression pour les bases de données statistiques. Par exemple, les études ont montré l'intérêt de ne pas stocker les valeurs des catégories, mais simplement un code. Des travaux ont également porté sur l'utilisation de matrices de stockage multidimensionnelles. Les techniques de linéarisation de tableaux sont alors utilisées pour optimiser le stockage physique des données. Cependant, le problème de faible densité apparaît dès lors que l'on choisit une représentation en tableau multidimensionnel. Toutefois, les valeurs nulles ont tendance à se regrouper, et la construction de séquences de valeurs nulles optimales (au sens des algorithmes de compression) est alors possible.

La manipulation des bases de données statistiques a été étudiée notamment dans [BRA 81]. Dans ce travail, l'auteur décrit les fonctionnalités du langage de manipulation proposé. Un système est proposé, SAS (*Statistical Analysis System*), qui fournit des manipulations spécifiques, comme par exemple l'échantillonnage, la possibilité pour l'utilisateur de définir ses propres fonctions, ou l'utilisation de résultats d'un traitement comme entrée d'un autre. Comme nous l'avons vu précédemment, [GHO 88] propose d'étendre l'algèbre relationnelle classique.

[CHA 81] propose une interface permettant l'exécution de requêtes fondée sur des menus de commandes (mais pas de langage).

## 1.2 Traitement des données

La construction de systèmes alliant gestion de bases de données et outils d'analyse statistique n'est pas aisée. Trois solutions sont possibles, consistant soit à enrichir les outils d'analyse statistique avec des outils de gestion de bases de données, ou l'inverse [IKE 81], ou encore à interfacer des outils d'analyse statistique et des systèmes de gestion de bases de données.

Dans [IKE 81], les auteurs considèrent que les systèmes de gestion de bases de données classiques nécessitent l'introduction de différentes fonctions statistiques : différenciation des valeurs continues et discrètes, gestion des données manquantes, calcul du nombre d'enregistrements pour une valeur donnée, de la précision, du maximum, du minimum, de la médiane, de la moyenne et des moments d'ordre deux, trois et quatre, etc. Des tables résumés sont construites, et des opérations, comme la projection par exemple, sont fournies pour explorer les données ainsi représentées.

Les traitements statistiques souhaités sont de plusieurs types : la vérification de données, l'exploration, ou la confirmation [BAT 82]. La vérification de données consiste notamment à découvrir des exceptions dans la base. L'exploration vise à décrire la distribution des variables et leurs relations. Enfin, la confirmation consiste à valider ou invalider des hypothèses (souvent issues de la phase d'exploration) en les déployant sur les données.

Les aspects liés à la sécurité sont également étudiés. Ils concernent notamment le souhait de fournir une information statistique sur les données et d'autoriser le plus de requêtes possibles sans pour autant dévoiler des informations confidentielles sur les individus. Un nombre très important de travaux de la littérature existe sur ce sujet. Différentes techniques sont possibles pour augmenter la sécurité des bases, notamment par perturbation des valeurs, ou par restriction des requêtes autorisées [KAM 77]. Par exemple, les valeurs sont modifiées, soit au moment de l'affichage des résultats, soit au sein de la base elle-même, l'erreur introduite devant alors rester dans des limites acceptables. La composition de groupes d'individus indivisibles est également possible. On ne peut plus alors retrouver des informations concernant les individus qui concernent le groupe, seules des données pré-agrégées sont disponibles. Cependant l'utilisateur peut retrouver des valeurs individuelles correspondant à des requêtes illégales (visant à retrouver des caractéristiques confidentielles d'un individu de la base) en combinant les résultats de requêtes légales. [SHI 99] souligne ce problème et propose un algorithme efficace de détection des requêtes illégales.

De nombreux systèmes implémentés ont été proposés, par exemple SUBJECT, décrit dans [CHA 81].

## 1.3 Vers l'OLAP et les bases de données multidimensionnelles

Le contexte OLAP (*On-Line Analytical Processing*), apparu en 1993 et détaillé dans le chapitre suivant, a repris de nombreux concepts des bases statistiques.

Les relations du contexte OLAP avec les bases statistiques ont été étudiées, notamment par [ROT 96, SHO 97, LEN 97], mettant en évidence les similarités de ces deux modèles. Les concepts communs à ces deux approches sont la multidimensionnalité, les hiérarchies et des attributs de résumés (appelés mesures dans le modèle OLAP). Dans les deux contextes, on trouve des données de base (également appelé *niveau micro*), des données de niveau agrégé (également appelé *niveau macro*), et des méta-données (hiérarchies par exemple). Les données sont distinguées selon qu'elles sont des *mesures* ou des *variables*. Si les bases statistiques ne présentent souvent à l'utilisateur que les données de niveau macro, les outils OLAP tendent à travailler à partir des données de niveau micro. Sur le plan des modèles conceptuels, les bases de données statistiques utilisent des modèles de graphes et tabulaires quand les modèles OLAP utilisent des modèles de cube. Au niveau du stockage physique des données, les modèles OLAP multidimensionnels et les systèmes de bases de données statistiques utilisent tous deux les techniques de linéarisation de tableaux. Cependant, il reste que les outils OLAP ne gèrent pas le problème de la protection de la vie privée puisque les données individuelles sont la plupart du temps disponibles.

Les travaux portant sur les bases de données statistiques, notamment reportés dans les actes de la conférence *Statistical and Scientific Database Management* (SSDBM), constituent un point de départ à ne pas négliger pour l'étude des bases multidimensionnelles et du cadre OLAP, que nous détaillons dans le chapitre suivant. L'ensemble des travaux proposés dans le cadre des bases de données statistiques concernent des aspects importants de l'approche OLAP qui lui succède :

- nature multidimensionnelle des données,
- importance des interfaces utilisateurs,
- navigation dans les données à différents niveaux d'agrégation,
- utilisation d'outils d'analyses statistiques,
- réduction de la dimensionnalité et évaluation des différentes représentations en fonction de leur utilité pour l'analyse,
- gestion de gros volumes de données,
- représentation physique de données à faible densité,
- hétérogénéité des données sources,
- ...

# Chapitre 2

# Bases de données multidimensionnelles

Les données de production des entreprises sont souvent stockées dans des bases de données relationnelles (voir annexe B). Cependant, cette architecture n'est pas toujours adaptée dans le cadre d'applications décisionnelles d'analyse en ligne des données (OLAP, *On-Line Analytical Processing*). Pour de telles applications, l'utilisateur a besoin d'information résumée, agrégée, et non d'outils de manipulation des informations sur les individus constituant la base. Ceci requiert l'application d'opérations de jointures très coûteuses.

Par exemple, dans un cadre commercial, l'analyste ne veut pas savoir quelles sont les meilleures ventes mais plutôt quel est le type des meilleures ventes sur une période donnée, pour une ville précise et un groupe d'âge ; ou encore il souhaite savoir ce qui se passerait si une action était tentée. On appelle cette deuxième forme de requête les requêtes *What If.*

Dans les bases de données décisionnelles, l'intérêt de l'utilisateur ne se situe donc plus au niveau des individus mais plutôt au niveau de l'identification de tendances dans un ensemble ou un groupe. Les bases de données relationnelles, optimisées pour la production et la mise à jour de données, selon un modèle de traitement OLTP (*On-Line Transaction Processing*), deviennent alors inadéquates pour le traitement à la volée sur de nombreux enregistrements dans le cadre de requêtes complexes nécessitant l'évaluation de nombreux agrégats.

Le traitement OLTP, tout à fait adapté aux requêtes simples, atteint donc ses limites quand il s'agit de traiter de très grosses bases de données dans une optique décisionnelle, avec des requêtes plus complexes, nécessitant des calculs importants sur un grand nombre de données. C'est dans ce cadre que sont apparus les entrepôts de données (DataWarehouses) pour traiter de gros volumes de données issues de sources hétérogènes à des fins d'analyse.

## 2.1 Entrepôts de données

Le concept d'entrepôt de données a été proposé par W.H. Inmon en 1990 pour répondre aux besoins d'analyse de données que les systèmes transactionnels ne pouvaient assurer [INM 96].

### 2.1.1 Définition

Les entrepôts sont définis comme une *collection de données servant un processus décisionnel.* De manière plus complète, [INM 96] propose la définition suivante :

*A data warehouse is a subject-oriented, integrated, time-variant, and non-volatile collection of data in support of management decision making process.*

Les données collectées concernent donc toutes un *même thème* et non toutes les opérations en cours d'une entreprise, leurs différentes sources sont *rassemblées* et *fusionnées* au sein d'un ensemble cohérent, elles sont *identifiées par une date*, et elles sont *maintenues* au sein de l'entrepôt (des ajouts sont possibles, mais aucune suppression n'intervient).

Cette définition est encore valable dans les systèmes actuels, cependant on note que les collections de données visant un unique sujet d'analyse sont plutôt des *magasins de données*, tandis que les entrepôts eux regroupent des données plus variées reflétant l'ensemble de l'activité d'une entreprise.

De plus, les volumes de données, pouvant atteindre plusieurs tera-octets, ne permettent pas la conservation de tout l'historique, et des données peuvent donc être effacées.

Ralph Kimball propose une définition plus simple [KIM 96] :

*A datawarehouse is a copy of transaction data specifically structured for query and analysis.*

Les entrepôts de données sont couramment définis comme des vues matérialisées de données historisées stockées à des fins d'analyse. Ils doivent répondre à un certain nombre de problèmes, liés notamment à la mise à jour efficace des données face au flux constant de données hétérogènes produites dans les systèmes transactionnels, et liés également à la mise en œuvre d'applications décisionnelles.

### 2.1.2 Composants des entrepôts

L'architecture d'un entrepôt de données est divisée en couches successives de vues matérialisées de données [JAR 98].

Les *sources de données*, appelées également *bases opérationnelles* forment la première couche. Elles peuvent être structurées dans des systèmes de bases de données habituels, semi-structurées, ou non structurées et stockées dans des fichiers. Elles proviennent soit des données de l'entreprise (données *internes*), soit de données *externes*. Par exemple des fichiers provenant de l'INSEE sont parfois utilisés afin de caractériser des objets de la base (ainsi, on peut retrouver l'âge des personnes présentes dans la base à l'aide de leur prénom). Ces données sont souvent hétérogènes, c'est-à-dire que la même information peut être représentée de manières diverses, par exemple en utilisant plusieurs schémas de bases de données.

La couche centrale constitue la partie principale de l'entrepôt et correspond à une collection de données telles que décrites dans la section précédente. Elle est donc constituée de données intégrées, historisées, décrites à un fort niveau de détail.

Enfin, une couche *locale* contient des données fortement agrégées issues de la couche précédente, dédiées à l'analyse. Différents types de couches locales sont possibles, comme par exemple les *magasins de données*. Ces magasins de données sont des petits entrepôts constitués d'un sous-ensemble des données correspondant à un sujet précis, rendant très rapides les temps de réponses aux requêtes.

Un niveau intermédiaire peut être utilisé, correspondant à un ODS (*Operational Data Store*) [INM 95, IMH 98, INM 00]. Il constitue alors une couche se situant entre les données sources et le niveau principal. Il est constitué de données portant sur un même sujet, non historisées, intégrées, volatiles, stockées à un niveau détaillé. Même si dans les deux cas les données sources sont intégrées et sont rassemblées autour d'un même sujet, cette approche se distingue de la couche principale des entrepôts en ce que les données sont mises à jour et non historisées. Ainsi, si les données d'un entrepôt ne sont souvent mises à jour au mieux que toutes les 24 heures environ, les données d'un ODS sont actualisées de manière continue. La fréquence des mises à jour n'est pas si cruciale dans les entrepôts de données qui contiennent des données dites *résumées statiques* car leur pertinence ne se détériore pas très rapidement. L'entrepôt est utilisé pour prendre des décisions au vu de données historisées sur une longue période, et non pas fondée sur la seule connaissance d'une donnée qui peut avoir changé depuis la dernière mise à jour.

### 2.1.3 Construction

La construction de la couche principale de l'entrepôt à partir des sources hétérogènes de données est un problème crucial et très complexe mettant en œuvre des aspects liés :

- à l'analyse (détection des valeurs non valides),
- aux méta-données (conception de l'entrepôt de données),
- au nettoyage (gestion des inconsistances des données sources),
- à l'extraction (accès aux différentes sources),
- à la transformation (formats de données etc),
- au chargement.

Les outils effectuant les phases d'extraction, transformation et chargement sont les outils d'*ETL* (*Extract, Transform, Load*), souvent rassemblés sous les termes d'*adaptateurs* (*adapters* ou de *wrappers*) et de *chargeurs* (*loaders*) [JAR 98]. Les *médiateurs* rassemblent les données dans l'entrepôt et résolvent les conflits liés à l'utilisation de plusieurs sources. L'acquisition des données pour la construction de l'entrepôt est très coûteuse et peut demander des mois à s'exécuter. De plus, l'entrepôt doit être constamment maintenu à jour (par exemple tous les mois).

Le nettoyage des données (*data cleansing*[1] ou *data scrubbing*) fait partie de la phase de transformation. Des données fausses sont non seulement inutiles, mais surtout dangereuses dans le cadre d'applications décisionnelles, une grande attention est donc requise lors de la phase de nettoyage. Cette phase est donc celle qui vise à évaluer et garantir la qualité des données. La problématique est très complexe, le nettoyage devant notamment se prémunir face à [JAR 98] :

- l'hétérogénéité des formats utilisés pour décrire les données (par exemple une ville décrite soit par son code postal soit par son nom),
- l'incohérence de données due à des fautes de frappe,
- la présence de données manquantes,
- la duplication de certaines données.

L'étape de transformation comprend également une phase d'agrégation des données. Le choix du niveau d'agrégation au moment de la construction de l'entrepôt est crucial puisque les données de niveau détaillées sont perdues après agrégation. Cependant, cette

[1] Le terme *data cleaning* est également utilisé.

perte est compensée par des temps de réponse aux requêtes qui sont d'autant plus courts que les données sont agrégées.

Les méta-données sont stockées et utilisées tout au long de l'utilisation de l'entrepôt, depuis la conception et la construction jusqu'aux phases d'analyse.

Les entrepôts, une fois construits, sont utilisés à des fins d'analyse des données. Cette problématique est détaillée dans la section ci-dessous.

## 2.2 Approche OLAP

Le terme OLAP (*On-Line Analytical Processing*) a été proposé en 1993 par E.F. Codd [COD 93]. Il désigne une catégorie d'applications et de technologies permettant de collecter, stocker, traiter et restituer des données multidimensionnelles à des fins d'analyse. En 1995, Nigel Pendse et Richard Creeth de l' *OLAP Report* ont introduit un autre terme pour décrire ce type d'outils, et ont proposé l'acronyme **FASMI** signifiant **F**ast **A**nalysis of **S**hared **M**ultidimensional **I**nformation [PEN 95a, PEN 95b]. Ce terme est traduit par *Analyse Rapide d'Information Multidimensionnelle Partagée.*

- **Analyse :** le système doit fournir des outils d'analyse numérique et statistique, soit prédéfinis par les programmeurs, ou définis de manière *ad-hoc* par l'utilisateur.
- **Rapide :** le système doit être conçu pour répondre aux requêtes de l'utilisateur de manière rapide (moins de cinq secondes), même si un calcul peut impliquer des millions de données[2]. Les techniques de stockage, d'architecture matérielle et de prétraitement influencent les temps de réponse. Actuellement, aucun produit disponible n'est encore complètement optimisé, même si les techniques de précalculs font tendre les performances vers celles souhaitées.
- **Information :** le système doit pouvoir accéder à l'ensemble des données, quel que soit leur volume et leur mode de stockage.
- **Multidimensionnelle :** Comme dans tout système OLAP, un système FASMI doit fournir une vue multidimensionnelle des données, et doit donc notamment gérer les hiérarchies (simples ou multiples).
- **Partagée :** Les données doivent être sécurisées en vue d'une architecture multi-utilisateurs.

Les bases de données multidimensionnelles permettent donc d'analyser, de synthétiser les informations et représentent une alternative très intéressante aux systèmes relationnels classiques, qui sont eux destinés au traitement individuel des données sans optique d'analyse ou de décision via des traitements OLTP.

Les différences entre traitements OLAP et OLTP sont résumées dans le tableau 2.2. Les avantages des bases multidimensionnelles sont principalement les suivants :

- gestion optimisée des agrégats,
- navigation aisée le long des hiérarchies de dimensions,
- visualisation et manipulation intuitives de manière multidimensionnelle.

Contrairement aux bases de données relationnelles, aucun modèle consensuel de bases de données multidimensionnelles n'existe, et différents auteurs ont proposé des modèles.

[2] Une étude indépendante hollandaise a estimé qu'un utilisateur considère qu'un processus est trop long à répondre si le temps de réponse dépasse 30 secondes [PEN 95b]. Au delà, l'utilisateur pense que le processus a échoué et est tenté de presser les touches *Ctr-Alt-Sup* pour le tuer, à moins que le système n'ait prévenu qu'un temps plus long était nécessaire. Cependant, même s'il a été prévenu, l'utilisateur perd sa concentration et le fil de son analyse, qui en souffre donc.

| | OLAP | OLTP |
|---|---|---|
| Type de requêtes | complexes | simples |
| But | optique décisionnelle | production et mise à jour des données |
| Niveau de vision | vision ensembliste (tendances ...) | vision au niveau individuel |
| Utilisateurs | analystes et décideurs (peu nombreux) | agents opérationnels (nombreux) |

TAB. 1 – OLAP *vs* OLTP

Les principaux modèles sont décrits dans [AGR 97], [GYS 97], [LEH 98], [GRA 97], [CAB 97, CAB 98], [VAS 98], [HAR 96], et [LI 96].

Des comparaisons ont été faites [VAS 99], [CHA 97], [BLA 98], et des initiatives telles que le *OLAP Report*[3] de *Business Intelligence* étudient les différents systèmes existants.

D'une manière générale, les données sont stockées dans des (hyper)cubes, sur lesquels plusieurs opérations sont définies.

## 2.3 Cubes de données

Un cube est un ensemble de données organisées selon des dimensions. On appelle *mesure* ou *élément* la valeur contenue dans une cellule du cube, associée aux valeurs (positions) prises sur les dimensions composant le cube.

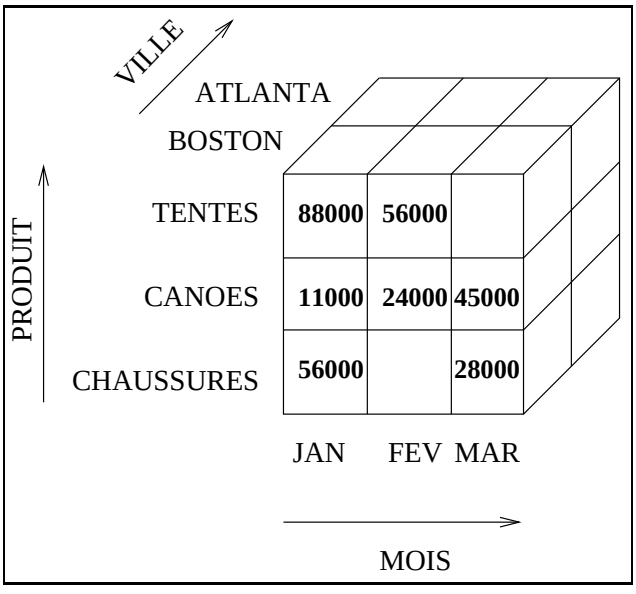


FIG. 2 – Exemple de cube

Un exemple de cube est donné par la figure 2. Il s'agit d'un cube à trois dimensions : *PRODUIT, MOIS,* et *VILLE*. La *mesure* du cube est constituée par les résultats des ventes

[3] `http ://www.olapreport.com`

de ces produits pour ces différentes villes à différents mois. Ainsi, il s'est vendu 88000 unités de *TENTES* dans la ville de *BOSTON* au mois de *JANVIER*. Ici, on voit bien que l'on a perdu le niveau de détail de chacune des transactions et que l'information visualisée a subi une première agrégation.

Nous présentons maintenant les principaux modèles existants, en distinguant les modèles purement multidimensionnels des modèles issus des bases de données relationnelles.

Les principales approches de la gestion des hiérarchies dans les modèles multidimensionnels proposés sont également présentées. La notion de dimension est primordiale dans les bases de données multidimensionnelles, notamment pour la représentation de plusieurs niveaux de granularité et la navigation entre ces niveaux. On distingue alors les dimensions plates des dimensions hiérarchiques en introduisant les niveaux de granularité. Ceux-ci permettent d'analyser les données à plusieurs niveaux. Ils sont très importants puisqu'ils affectent d'une part le volume de la base et, d'autre part, le niveau de requête que l'on pourra déployer.

Dans l'exemple considéré précédemment, la dimension *MOIS* peut très facilement être hiérarchisée, en ajoutant par exemple un niveau inférieur pour les semaines et un niveau supérieur pour les années. On pourra alors naviguer entre niveaux, remonter au niveau supérieur année ou redescendre dans la hiérarchie jusqu'au niveau semaine.

Les hiérarchies de dimensions ne sont pas toujours prises en compte dans les modèles existants et ne sont pas toujours modélisées explicitement.

## 2.4 Opérations

Les opérations algébriques définies pour la manipulation des cubes ne sont pas encore uniformisées mais reposent souvent sur les primitives présentées dans [CHE 96], [AGR 97], [HAN 97]. On distingue deux types d'opérations : les opérations *inter-cubes*, telles que Jointure, Union, Intersection et Différence, et les opérations *intra-cubes* qui sont de deux types : celles qui modifient le contenu du cube et celles qui ne changent que la représentation.

Les principales opérations sur la structure des cubes sont : la rotation (*Rotate*), l'inversion (*Switch*), la décomposition (*Split*), l'imbrication (*Nest*) et la concaténation (*Push*).

Les principales opérations sur le contenu des cubes sont : la généralisation (*Roll-Up*), la spécialisation (*Drill-Down*), la restriction (*Slice*) et la projection (*Dice*).

### 2.4.1 Opérations sur la structure des cubes

#### Rotation

La rotation (figure 3) permet d'examiner le cube selon un autre angle, sachant qu'un cube de dimension $n$ a $n(n-1)$ vues (par exemple un cube de dimension 3 a 6 faces).

#### Inversion (*Switch*)

Cette opération consiste à inverser l'ordre de certaines valeurs (figure 4). Certaines informations se retrouvent alors placées l'une à côté de l'autre, ce qui facilite la découverte d'un phénomène.

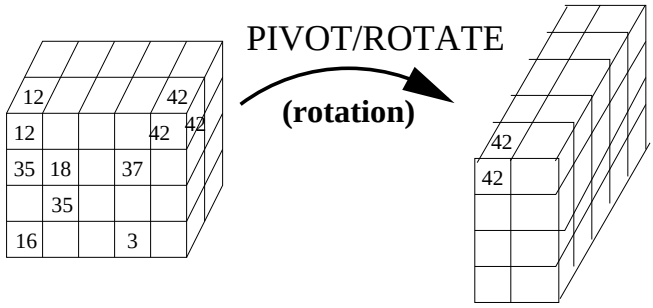


FIG. 3 – rotation

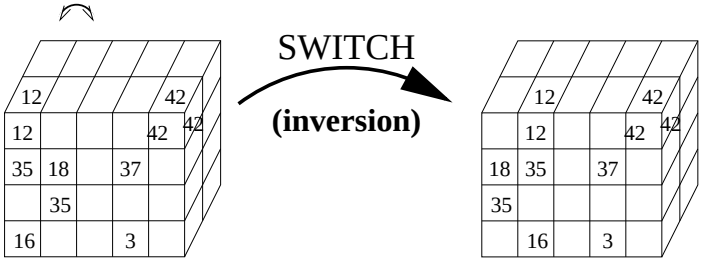


FIG. 4 – Inversion

### Décomposition (*Split*)

La décomposition consiste à décomposer les données selon certaines valeurs de dimensions. Le cube est alors représenté par un ensemble de sous-cubes dont chacun correspond à une valeur de la dimension considérée. Ceci permet donc de réduire le nombre de dimensions. Sur l'exemple de la fig. 2, le cube des ventes peut être décomposé en 3 cubes décrivant les ventes de (1) tentes, (2) canoës et (3) chaussures.

### Imbrication (*Nest*)

L'imbrication (figure 5) permet de faire rentrer une dimension à l'intérieur d'une autre (Nest) ou de l'en sortir (Unnest).

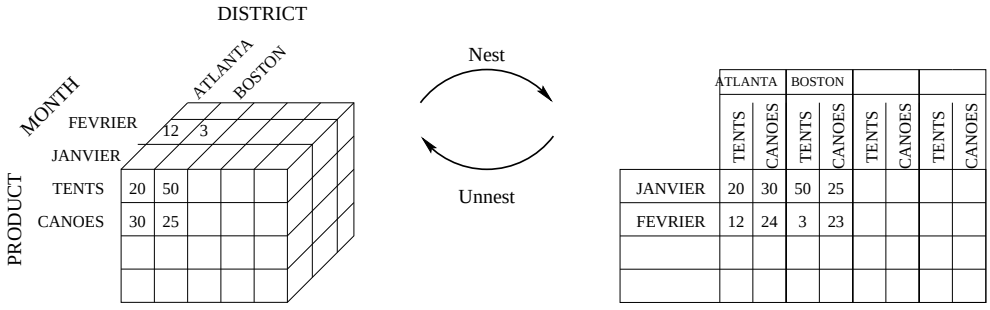


FIG. 5 – Imbrication

### Concaténation (*Push*)

La concaténation (figure 6) fait passer les valeurs des cellules dans d'autres cellules (Push) ou les en sort (Pull).

Les cellules obtenues par l'opération de Push sont donc des n-uplets alors que l'opération Nest conserve toutes les dimensions, tout en en cachant une.

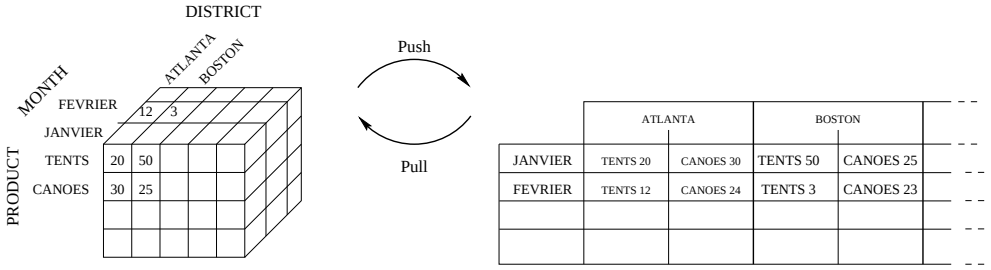


FIG. 6 – Concaténation

## 2.4.2 Opérations sur le contenu des cubes

### Généralisation (*Roll-Up*)

Le roll-up (figure 7) revient à généraliser les valeurs d'un attribut à des concepts de niveau supérieur (passage au granule supérieur). On peut par exemple calculer la moyenne sur plusieurs valeurs de dimensions. Le roll-up (*) ou roll-up global calcule l'agrégation sur toutes les valeurs, on n'a plus qu'une cellule contenant, par exemple, la moyenne générale[4].

La spécification (*drill-down*), à l'inverse du roll-up, est une opération de spécialisation (passage au granule inférieur).

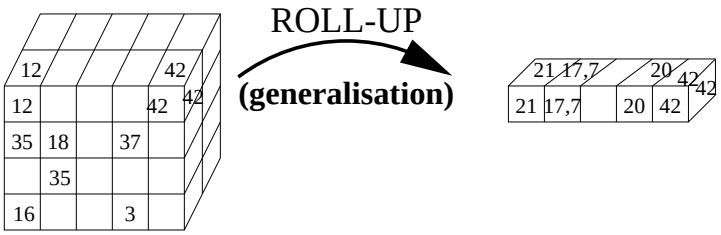


FIG. 7 – Roll Up

### Sélection sur les cellules (*Slice*)

La restriction (figure 8) consiste à extraire de l'information résumée pour une certaine dimension.

On ne retient alors que les valeurs correspondant à un certain critère, par exemple toutes les valeurs inférieures à 15.

[4]La fonction d'agrégation peut tout aussi bien être la somme, la médiane ... ou une fonction plus complexe

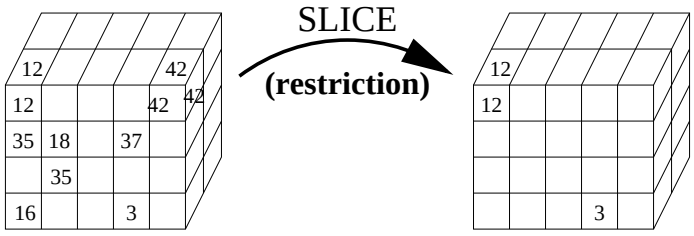


FIG. 8 – Sélection sur les cellules

**Sélection sur les dimensions (*Dice*)**

Il s'agit d'une restriction sur les dimensions et non plus par rapport à un critère sur la mesure (voir figure 9).

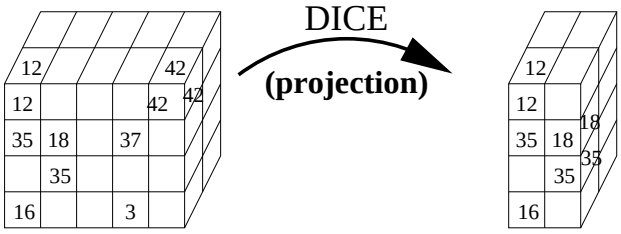


FIG. 9 – Sélection sur les dimensions

## 2.5 Modèles MOLAP et ROLAP

Les modèles multidimensionnels sont construits soit sur un modèle relationnel, soit sur un modèle physique réellement multidimensionnel.

La première approche, l'approche ROLAP (*Relational* OLAP), permet l'utilisation des systèmes de gestion de bases de données relationnels, très performants et robustes face aux gros volumes de données. Cependant, ces systèmes sont optimisés pour la gestion de bases normalisées dans une optique de bases de production, et répondent difficilement aux enjeux liés aux bases décisionnelles.

La seconde approche, l'approche MOLAP (*Multidimensional* OLAP), optimise les temps de réponses aux requêtes, les données étant physiquement matérialisées de manière multidimensionnelle. Cependant, les systèmes sont nouvellement construits et ne sont pas robustes face aux très gros volumes de données.

Certains systèmes couplent ces deux approches dans des systèmes HOLAP (*Hybrid* OLAP). Des études comparatives ont été menées sur ces trois types de bases (par exemple dans [FIR 97] et [SON 00]), et des modèles ont été proposés pour chacun.

Les systèmes MOLAP utilisent une base de données multidimensionnelle dans laquelle tout le cube est physiquement matérialisé. Cette méthode conduit à des temps de réponse aux requêtes très rapides, cependant, elle est très coûteuse en terme d'espace mémoire de stockage.

Les systèmes ROLAP stockent, eux, les données dans une base de données relationnelle. Le cube n'est pas matérialisé sauf au moment de la phase de requête.

Enfin, dans les systèmes HOLAP, seule une partie du cube est matérialisée de manière optimisée sous forme multidimensionnelle. Les autres données sont laissées dans la base

relationnelle et extraites de manière dynamique au moment des requêtes.

La distinction entre ces trois matérialisations est due au fait que le stockage physique des cubes, s'il augmente les performances des requêtes, est très coûteux en terme de place. Ainsi, la méthode MOLAP permet des réponses aux requêtes plus rapides, mais ne peut être utilisée pour de gros volumes de données. La méthode HOLAP présente quant à elle un compromis entre optimisation de stockage et temps de réponse.

Les bases MOLAP utilisent des formats de stockage et d'accès aux données propriétaires alors que les bases relationnelles bénéficient des standards définis, tant pour le modèle de représentation, que pour les techniques d'indexation et de manipulation (y compris le langage de requête SQL). Cependant, pour une analyse multidimensionnelle, les utilisateurs ne disposent pas de véritables langages de requête, et doivent utiliser des langages propriétaires, souvent sous forme de bibliothèques de fonctions (*Application Program Interface* - API).

### 2.5.1 Modèles MOLAP

Les modèles MOLAP stockent les données dans des tableaux multidimensionnels, et fournissent de très bonnes performances pour les temps de réponse aux requêtes, les données étant matérialisées de manière multidimensionnelle, et les opérations sur les hiérarchies ne nécessitant aucune opération coûteuse de jointure. Des opérateurs OLAP sont fournis de manière native, sans recours à des combinaisons complexes d'opérateurs SQL existants comme dans le cas du ROLAP. Les données étant agrégées, les volumes sont plus faibles que dans les bases relationnelles, et l'index est donc plus facile à gérer (la plupart du temps, il tient en mémoire). Les problèmes liés à cette représentation sont principalement de trois types :

- les mises à jour de données nécessitent de lourds changements,
- les tableaux multidimensionnels sont souvent très creux, cependant de nombreux algorithmes de gestion de matrices creuses existent pour pallier cet inconvénient,
- aucun modèle consensuel n'existe, ni pour représenter ni pour interroger ces données stockées de manière multidimensionnelle.

#### Avantages de la représentation multidimensionnelle : exemples

On donne ici deux exemples pour mieux cerner les avantages de la représentation multidimensionnelle par rapport aux bases relationnelles. Ces exemples sont adaptés à partir des informations du site `http ://server.lsol.tm.fr/clubfr/event/ats_004`.

On reprend l'exemple de la figure 2, en considérant que chaque dimension peut prendre 10 valeurs. Les données se présentent alors sous la forme d'un cube $10 \times 10 \times 10$. L'équivalent sous la forme relationnelle est une table de 1000 lignes de 4 champs.

Supposons que nous voulions connaître les ventes de Chaussures réalisées dans la ville X. Avec le modèle relationnel, la recherche peut se faire sur les 1000 lignes de la table. Avec le modèle multidimensionnel, la recherche s'effectue parmi les 3 dimensions de 10 positions pour trouver la bonne cellule. En supposant que la recherche moyenne soit égale à la moitié de la recherche totale, le nombre d'itérations est de 15 avec le modèle multidimensionnel contre 500 avec le modèle relationnel.

Supposons maintenant que nous voulions connaître les ventes totales de chaussures. Le modèle relationnel suppose la lecture totale des enregistrements alors qu'avec le modèle multidimensionnel, il suffit d'agréger une *tranche* de $10 \times 10$ valeurs.

On note que le modèle multidimensionnel n'est performant que si les données représentées sont fortement liées, sinon les cubes construits sont très peu denses (le nombre de cellules contenant une valeur est très faible), même si les vendeurs d'applications OLAP (Oracle et Microsoft par exemple) optimisent le stockage et la restitution des données sur des cubes peu denses.

### Modèles de la littérature

Selon le modèle d'Agrawal, un cube a les composantes suivantes [AGR 97] :
- $k$ dimensions notées $D_i$ $(i = 1, \ldots k)$, chacune d'elles étant associée à un domaine de valeurs noté $dom_i$ $(i = 1, \ldots k)$,
- des éléments $E(C)(d_1, \ldots, d_k)$, appartenant à un ensemble $V$ des valeurs du cube, où $d_i \in dom_i$ pour $i = 1, \ldots, k$. Ces éléments peuvent être soit un n-uplet, soit 0 ou 1. Le cas 0 signifie que la combinaison des valeurs de dimensions n'existe pas dans la base. Dans le cas où l'on a la valeur 1, alors il existe une combinaison, et dans le cas où l'élément est un n-uplet, alors on dispose d'une information complémentaire pour cette combinaison de valeurs de dimensions.

Le modèle repose donc sur le modèle d'hypercube avec un ensemble d'opérations. Les opérations sont closes, ce qui signifie que l'application d'une opération sur un cube produit un cube. On peut ainsi composer plusieurs opérations. Les dimensions et les mesures sont traitées de manière symétrique. Ainsi, les sélections et les agrégations sont autorisées sur toutes les dimensions et les mesures.

Les hiérarchies ne sont pas décrites de manière explicite par le modèle, et des fonctions de fusion de dimension sont considérées pour réaliser les agrégations.

[CAB 98] introduit un modèle de base de données multidimensionnelle $\mathcal{MD}$ indépendant de l'implantation logicielle (relationnelle ou multidimensionnelle). Dans ce modèle, une base multidimensionnelle est représentée par les notions de dimensions et de *f-tables*. Les dimensions sont construites à partir de niveaux de hiérarchie, d'un ordre partiel et d'un ensemble de fonctions d'agrégation. Le schéma multidimensionnel est alors obtenu par un ensemble fini de dimensions, un ensemble de *f-tables* et un ensemble fini de descriptions de niveaux. Les *f-tables* stockent les données factuelles. Chaque *f-table* est de la forme $f[A_1 : l_1\langle d_1\rangle, \ldots, A_n : l_n\langle d_n\rangle] : l_0\langle d_0\rangle$ où $f$ est un nom, chaque $A_i$ $(1 \leq i \leq n)$ est un nom différent appelé *attribut* de $f$ et chaque $l_i$ $(0 \leq i \leq n)$ est un niveau de la dimension $d_i$.

Les hiérarchies sont représentées sous forme de treillis, avec une famille de fonctions de généralisation définissant comment s'opèrent les fusions de données au cours de la navigation entre les différents niveaux de la hiérarchie.

Dans son modèle, Vassiliadis représente de manière explicite les dimensions et les hiérarchies de dimensions [VAS 98]. Une algèbre pour les opérations OLAP est fournie. Le modèle est fondé sur le concept de *cube de base*, qui représente les données au niveau le plus détaillé. Tous les autres cubes sont calculés à partir de ces cubes de base. Les hiérarchies sont représentées sous forme de treillis.

Dans le modèle proposé par [LEH 98, ALB 99b, GUE 99], les dimensions sont hiérarchisées et représentées par différents types d'attributs. Les attributs primaires sont utilisés pour représenter le niveau le plus détaillé des informations, et les attributs de classification

sont utilisés pour représenter les hiérarchies le long des dimensions, sous forme arborescente. Cette structuration des dimensions ne permet donc pas de modéliser des hiérarchies telles que celle de la figure 10.

Le modèle de cube proposé repose alors sur les objets multidimensionnels primaires et secondaires. Un *Objet Multidimensionnel Primaire*, qui représente un cube, consiste en un identifiant de cellule, un schéma de définition, un ensemble de sélections, un type d'agrégation (somme ou moyenne par exemple) et un type de résultat. La détermination du type d'agrégation permet de spécifier les types d'agrégations supportés par les données. Les différents types d'agrégation possibles sont $\Sigma$ (les agrégations de type somme sont acceptés sur cette dimension), $\phi$ (les agrégations de type moyenne sont acceptés sur la dimension considérée) et $c$ (les données sont de type *constantes* et donc seules les agrégations de type comptage sont acceptables sur la dimension considérée).

Un *Objet Multidimensionnel Secondaire* est constitué de tous les niveaux de dimension pour lesquels on peut généraliser ou spécifier (opérations de navigation à travers les niveaux de granularité).

Le système CUBESTAR est l'application logicielle du modèle. Dans ce système, les données sont physiquement stockées sous forme relationnelle, même si les requêtes sont toutes de nature multidimensionnelle.

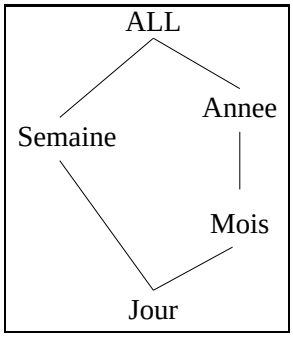


FIG. 10 – Exemple de hiérarchie multiple

### 2.5.2 Modèles ROLAP

#### Caractéristiques de l'approche ROLAP

L'idée sous-tendant l'approche ROLAP est d'étendre les systèmes de gestion de bases de données relationnels à l'approche OLAP, en incluant donc :

- une vision multidimensionnelle des données,
- le calcul de données agrégées,
- les opérations de navigation entre niveaux de hiérarchies,
- la proposition de requêtes SQL étendues aux besoins OLAP.

Des solutions sont proposées, fondées principalement sur les requêtes *group by* et les clauses *having*. Ces systèmes sont spécifiquement conçus pour la représentation multidimensionnelle, et sont organisés autour de deux types de données : les mesures (données numériques) stockées dans des *tables de faits*, et les dimensions, stockées dans des *tables satellites* et sont jointes aux tables de faits.

Afin de réduire les coûts de calculs des jointures, les tables sont *pré-jointes*, et les magasins de données sont organisés de manière *dénormalisée*. Le processus de mise sous forme

normale des bases de données, s'il réduit l'espace de stockage et interdit les redondances, n'est en effet pas adapté dans l'approche OLAP. Cependant, la dénormalisation des bases de données conduit aux mêmes problèmes que dans l'approche MOLAP précédemment présentée, c'est-à-dire notamment à l'explosion de l'espace de stockage nécessaire.

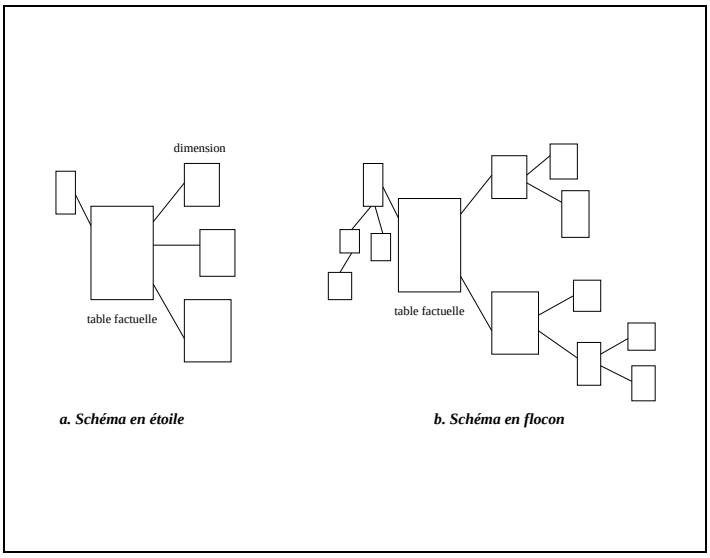


FIG. 11 – Schémas de bases relationnelles

Une solution à ce problème est l'organisation de l'information sous forme de schémas en étoile ou en flocon (voir figure 11), imposant une structure multidimensionnelle aux relations de la base. [LEV 02] expose les fondements théoriques liés à ce choix. Les schémas en étoile représentent de manière dénormalisée les dimensions. Par exemple, pour chaque ville, la région est répétée au sein de la même base, introduisant ainsi une dépendance fonctionnelle $ville \rightarrow region$. Les schémas en flocon sont donc proposés afin de ne pas avoir d'information redondante. Les dimensions sont alors représentées à différents niveaux de hiérarchie à travers plusieurs tables (au lieu d'une seule par dimension dans le schéma en étoile). Ces types de schémas simplifient l'interprétation des données, optimisent les performances des requêtes et nécessitent moins d'espace de stockage pour les très grandes bases.

### Modèles de la littérature

Le modèle de Gray et al. est une extension du modèle relationnel [GRA 97]. L'opérateur *data cube* est en effet une extension d'une table relationnelle, en calculant les agrégations de tous les sous-espaces possibles en combinant les attributs de la relation. Les auteurs ont introduit l'opérateur *CUBE* qui calcule toutes les agrégations marginales des données connues à un niveau de précision détaillé.

De même, le modèle proposé par Li et Wang est fondé sur les bases relationnelles [LI 96]. Les dimensions sont modélisées à l'aide de relations de dimensions (paires *(nom de dimension, nom d'attribut)*), et les cubes par des fonctions du produit cartésien de ces dimensions vers la mesure.

Pour les hiérarchies, on considère des relations et des partitionnements des données selon les valeurs d'attributs (sur le modèle du *group-by* SQL).

Par exemple on considère le schéma G ={Region,Storelocation} et la relation de regroupement

g =

| | |
|---|---|
| East | New York |
| East | Washington |
| West | Los Angeles |

Le modèle proposé dans [HAR 96] propose de ne matérialiser que les parties du cube nécessaires, c'est-à-dire que les données sont gérées sous forme relationnelle et ne sont représentées sous forme de cellules qu'en cas de nécessité. Le choix des cellules à matérialiser est donc le point prédominant de ce modèle. Les données multidimensionnelles sont gérées sous forme d'*hypercubes* construits le long de *dimensions*, dont la *mesure*. Les hiérarchies sont gérées par des treillis $(L, \preceq)$. Une dimension est alors constituée d'*attributs* $L$ ordonnés par un ordre partiel $\preceq$. On a par exemple $(année) \preceq (mois) \preceq (jour)$. Les hiérarchies *multiples* sont prises en compte, et sont distinguées des hiérarchies *simples*. Par exemple, les mois ne peuvent pas être décomposés en semaines (figure 10).

[BAR 97] reprend les mêmes idées, et considère qu'une base de données multidimensionnelle est une base de données qui fournit un environnement pour les analystes et les décideurs en supportant des requêtes complexes et faisant référence à des agrégations de données volumineuses historisées. Une base de données multidimensionnelle est alors considérée comme un entrepôt relationnel où l'information serait organisée en schémas en étoile.

Une telle base est donc composée d'un ensemble de tables $D_1, ..., D_n, F$ formant un schéma en étoile dénormalisé où chaque $D_i$ est une table de dimension et $F$ est la table de faits, c'est-à-dire une relation reliant tous les $D_i$. Des hiérarchies sur les dimensions sont modélisées grâce à l'introduction de dépendances fonctionnelles sur les attributs des tables de dimensions. Ces dépendances fonctionnelles doivent être acycliques, et on obtient alors une hiérarchie arborescente.

Le regroupement des attributs et la représentation sous forme de treillis des regroupements possibles sont également envisagés, et des algorithmes sont proposés pour optimiser les requêtes de regroupement nécessaires à de bonnes performances du système.

Le problème du choix des vues à matérialiser parmi le treillis de vues possibles a été repris par Ullman dans une présentation de la problématique lors de la conférence KDD [ULL 96].

[GYS 97] met l'accent sur le modèle conceptuel. La distinction est faite entre les aspects de structure et le contenu. Des bases sont définies sous la forme de tables multidimensionnelles. Les mesures complexes peuvent ainsi être représentées comme des attributs supplémentaires de la table de faits. Cette approche distingue les paramètres (attributs) et les mesures. Le modèle est équivalent à une représentation relationnelle classique.

Chaque hiérarchie est stockée comme un attribut séparé de la dimension. La représentation des hiérarchies est donc faite sur le modèle relationnel, avec un schéma en étoile ou en flocon [JAG 99]. La formalisation des hiérarchies est faite selon un domaine hiérarchique, un schéma hiérarchique (structure en arbre), et une hiérarchie. Des conditions pour la complétude et la cohérence sont données (par exemple on ne peut pas sauter de niveau). Une dimension est alors décrite par un nom associé à une hiérarchie.

### 2.5.3 Comparaison des modèles présentés

On reprend ici la comparaison effectuée dans [VAS 99], pour comparer les approches proposées par les différents auteurs. Cette comparaison est résumée par le tableau 2. Ce tableau détaille pour chacun des modèles la méthode de modélisation des cubes, la manière dont les hiérarchies sont représentées, et la nature du stockage physique. Les modèles sont distingués selon qu'ils sont orientés cube (cinq premiers modèles) ou orientés relationnel (cinq derniers modèles). La troisième propriété ne concerne que les modèles définis de manière directement multidimensionnelle, et indique une correspondance entre le modèle multidimensionnel et le modèle relationnel.

De plus, on peut identifier quelques différences fondamentales entre les modèles relationnels et multidimensionnels. Dans ce dernier,
- les dimensions sont organisées en hiérarchies (de manière explicite ou non),
- pour un cube donné, une (voire plusieurs) des dimensions est remarquable dans le sens où elle constitue la mesure (valeurs des cellules),
- l'ordre des dimensions dans la définition d'un cube est important (une opération de rotation est d'ailleurs définie).

| | Représentation des cubes (rep. physique) | Représentation des hiérarchies | Correspondance avec le relationnel |
|---|---|---|---|
| [AGR 97] | multidimensionnelle (cubes) | - | oui |
| [CAB 97, CAB 98] | multidimensionnelle (cubes) | explicite | oui |
| [VAS 98] | multidimensionnelle (cubes) | explicite | oui |
| [LEH 98] | multidimensionnelle (cubes) | explicite | implicite |
| [GRA 97] | relationnelle (tables) | - | |
| [LI 96] | relationnelle (tables) | implicite | |
| [GYS 97] | relationnelle (tables) | explicite | |
| [BAR 97] | relationnelle (tables) | explicite | |

TAB. 2 – Comparaison des modèles existants

## 2.6 Construction des cubes

La plupart des bases de données multidimensionnelles sont issues des entrepôts de données relationnels. Elles sont extraites depuis ces entrepôts vers des magasins de données multidimensionnels. Les données sont supposées extraites de systèmes opérationnels (par exemple la base de gestion des commandes d'une usine), validées, et résumées avant leur

incorporation dans un système OLAP. Cette étape est primordiale dans le processus, afin de s'assurer que l'information visualisée par l'utilisateur final du système OLAP est correcte et cohérente.

Les requêtes mettant en jeu des agrégations sont très importantes en OLAP, et les performances souhaitées doivent donc être optimales afin que le système respecte les prérequis de l'analyse *FASMI* et que l'analyste obtienne une réponse avec des délais de réponse raisonnables. L'équilibre entre optimisation des temps de réponse et du volume de données préagrégées stockées est crucial. En effet, si le système précalcule tous les agrégats possibles, la taille de la base explose. A l'opposé, s'il ne précalcule rien, les performances du système chutent.

Le problème de la construction de ces cubes à partir des bases initiales est donc un problème crucial pour l'analyse OLAP. Certains auteurs ont étudié ce problème et ont proposé des algorithmes pour optimiser le choix des cubes à matérialiser physiquement [HAR 96].

La *consolidation* consiste à préagréger les données en calculant par exemple la somme, la moyenne de chaque ligne et de chaque colonne, ou encore des cubes de contingence. Certains calculs coûteux sont ainsi effectués en amont et stockés dans les cubes pour être utilisés en décision, découverte de règles, etc. C'est un moyen de réduire les temps de réponse aux requêtes. Le type de fonction d'agrégation à mettre en œuvre doit donc lui aussi être défini à la création du cube.

Dans le cas où la consolidation n'est pas utilisée, on peut envisager des calculs à la volée pendant le traitement (notion de variable dérivée selon [PIL 95]), mais l'exécution est alors plus lente.

Le problème fondamental est l'explosion en terme de place des bases si les précalculs sont trop nombreux. Les exemples de cas réel illustrant ces effets abondent. Ainsi, un banc d'essai (*benchmark*) standard récemment publié fait état d'une explosion avec un facteur 240, nécessitant donc près de $2,4$ gigaoctets d'espace de stockage pour gérer une base de 10 mégaoctets en entrée.

En réponse à ces approches coûteuses, des systèmes se sont donc développés et déterminent quels sont les agrégats qui fournissent les meilleures améliorations des performances.

Les systèmes OLAP sont tous propriétaires et il n'existe pas de modèle ni d'implantation logicielle consensuelle. Les systèmes existants ont donc développé des techniques très proches de leurs applications.

Les systèmes tels SQL Server[5] prétendent gagner jusqu'à environ 80% de performance sans pour autant effectuer beaucoup de précalculs. Des heuristiques sont utilisées pour déterminer les précalculs intéressants desquels toutes les autres agrégations peuvent être dérivées de manière peu coûteuse. Ainsi, le choix judicieux des précalculs permet de générer tous les autres agrégats nécessaires à l'analyse sans avoir à parcourir la base à nouveau. Cependant, ces heuristiques sont fondées sur des modèles mathématiques qui peuvent correspondre ou non aux besoins de l'analyse. Un apprentissage en ligne des requêtes utilisateur permet donc de corriger des heuristiques selon la véritable utilisation du système.

Pour tout schéma relationnel de $n$ attributs, il existe $2^n$ cubes possibles à construire, comme l'illustre le treillis des dimensions (voir Fig. 12). Le nombre et la taille des cubes augmente de manière exponentielle avec le nombre de dimensions, et de manière linéaire avec la cardinalité des domaines de dimension.

[5] `http ://msdn.microsoft.com/library/backgrnd/html/olapover.htm`

[KIM 96] propose de stocker, si possible, tous les agrégats de toutes les combinaisons de dimensions possibles, cependant cette solution n'est pas réalisable en pratique, et la proposition est alors de ne matérialiser que les agrégats qui vérifient certaines propriétés de densité.

La construction des $2^n$ vues possibles est donc en pratique irréalisable. Des algorithmes se sont donc développés pour la construction partielle des cubes, y compris des approches parallèles [DEH 01].

Les *iceberg cubes* sont de tels cubes, construits à partir d'une contrainte d'agrégation *iceberg query* correspondant souvent à un seuil minimal d'occurrences [DES 97, GOI 98, FAN 98, FIN 00, CIC 01, HAN 01]. Ce problème est donc très proche des travaux concernant les motifs fréquents [AGR 93], et l'on cherche les combinaisons de valeurs d'attributs vérifiant la contrainte. Le terme *iceberg* a été choisi car les cubes construits ne contiennent qu'une infime partie des données (*partie émergée de l'iceberg*).

Pour chacune des possibilités dans le treillis des combinaisons d'attributs, les algorithmes de construction des iceberg cubes doivent déterminer s'il y a ou non des valeurs qui satisfont la condition. Deux approches opposées sont envisageables, soit descendante (*top-down*), soit ascendante (*bottom-up*). La première commence par calculer les partitions par groupements les plus agrégés possibles puis élargit les calculs à des groupements moins agrégés, tandis que la seconde, à l'inverse, commence à calculer les groupements les moins agrégés pour réduire ensuite aux groupements plus agrégés.

L'algorithme BUC (*Bottom-Up Computation*) adopte une approche ascendante pour le calcul de cubes partiels ou complets [BEY 99]. Les calculs débutent par les groupements les plus agrégés et se poursuivent de manière récursive pour les attributs suivants (passant donc à un cube moins agrégé). Le premier calcul est donc l'agrégation de toute la base. Chaque fois que la condition n'est pas satisfaite pour un calcul, l'algorithme stoppe les calculs pour les groupements suivants.

Les algorithmes TDC (*Top Down Computation*) et TDC-P (*Top Down Computation - Pruning*) avec élagage fonctionnent à l'inverse de BUC et débutent par les calculs avec les groupements les plus grands.

[FIN 00] compare les performances (temps d'exécution) de ces trois approches, les trois algorithmes produisant exactement les mêmes résultats. Les auteurs montrent que l'approche BUC est plus efficace que les autres approches, sauf dans le cas où il y a très peu d'attributs (TDC-P obtient alors des résultats comparables à ceux de BUC). TDC-P est meilleur que TDC.

[ULL 96, HAR 96] travaillent également sur le treillis des vues possibles d'un cube. Les auteurs présentent comment les requêtes (ou vues) sur les cubes peuvent être dépendantes. En effet, une requête $Q_2$ peut être utilisée pour répondre à une requête $Q_1$ et on dit alors que $Q_1$ dépend de $Q_2$ et on note $Q_1 \preceq Q_2$. Par exemple, les auteurs considèrent l'exemple d'une base à trois attributs *part (p), fournisseur (f), client (c)*. Il y a alors $2^3 = 8$ groupements possibles d'attributs. Comme précédemment, les requêtes (vues) possibles sont donc au nombre de 8 (*(pfc), (pc), (pf), (fc), (p), (f), (c), (∅)*). Ces vues sont considérées comme des requêtes avec une clause de regroupement *group-by*. Un algorithme d'optimisation de la matérialisation des cubes est proposé (*greedy algorithm*) pour ne pas avoir à construire tous les cubes matérialisés possibles.

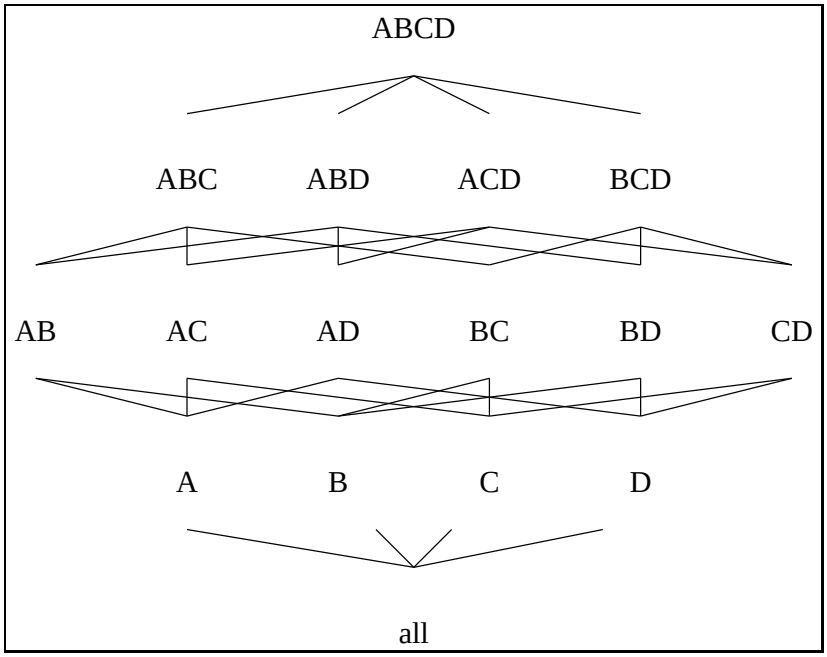


FIG. 12 – Treillis des cubes pour 4 dimensions

# Chapitre 3

# Bases de données floues

Les bases de données doivent préserver l'intégrité des données représentées. La prise en compte des données issues du monde réel nécessite donc l'extension des modèles classiques. En effet, les données issues du monde réel sont souvent imparfaites et les requêtes que les utilisateurs souhaitent mettre en œuvre souvent définies de manière vague.

Des travaux ont donc été menés dès les années 70 afin d'étendre les modèles de bases de données à la prise en compte de telles données. De nombreuses propositions ont été faites pour la prise en compte du flou dans les bases de données au niveau de la représentation des données et à celui des requêtes.

Ces travaux concernent tous les modèles possibles, depuis les modèles Entités-Relations jusqu'aux modèles objets et relationnels (voir [YAZ 92] ou [BOS 95a] pour une synthèse). Globalement, l'idée est d'assouplir à la fois les données contenues dans les tables de la base relationnelle en offrant la possibilité de stocker des données imprécises et en associant des degrés aux valeurs (dont les différentes sémantiques possibles seront exposées ci-dessous) et en autorisant les valeurs à être des distributions de possibilité, et d'assouplir les égalités par des similarités, par exemple dans le cas des dépendances fonctionnelles. Le dernier point concerne la gestion de requêtes flexibles.

L'introduction du flou dans les bases de données permet ainsi de représenter les données issues du monde réel. Il est en effet absolument nécessaire que la représentation des données soit conforme à leur nature, et que donc l'*intégrité* des données soit respectée. A. Motro dégage trois types d'imperfection dont les données issues du monde réel peuvent être entachées [BOS 95a] :

- Erreur
- Imprécision : l'information est fournie sous la forme d'un ensemble de valeurs plus ou moins possibles.
- Incertitude : qu'elles soient précises ou imprécises, nos connaissances ne peuvent parfois pas être affirmées avec certitude. Les informations imprécises ("l'âge de Paul est *31 ou 32*") sont distinguées des informations incertaines ("l'âge de Paul est *sans doute* 32"), même si ces deux notions sont liées. En effet, dans bien de cas, plus l'information est précisément décrite, moins on en est sûr.

Les outils mathématiques pour la gestion des imprécisions et des incertitudes sont rappelés en annexe A. Les sections suivantes présentent les différents modèles de bases de données floues. Un intérêt particulier est porté aux bases relationnelles floues car le modèle relationnel est très proche du modèle multidimensionnel que nous voulons étendre au flou.

## 3.1 Extension du modèle Entité-Relation au flou

Le modèle Entité-Relation est dû à Chen (1976). Il permet la représentation des informations du monde réel sous forme d'entités et de relations entre ces entités. Il s'agit d'un modèle conceptuel, souvent utilisé comme une première étape de modélisation du monde réel avant la transformation en schéma relationnel (voir section 3.3).

Les entités sont des ensembles de couples $(attribut, valeur)$ où les valeurs appartiennent au *domaine* de l'attribut. Les associations d'entités sont des relations entre deux ou plusieurs entités. Des contraintes de cardinalité sont utilisées pour spécifier, pour chaque type d'entité intervenant dans l'association, le nombre maximal et minimal d'associations dans lesquelles chaque entité du type considéré doit intervenir. Par exemple, on considère l'association *cpte_client(Client,0,N,Cpte,1,2)* où chaque client peut avoir autant de comptes qu'il le désire, et où chaque compte est attaché à un ou deux clients.

Une extension de ce modèle au flou a été proposée par Zvieli et Chen en 1985, avec l'introduction de trois niveaux de flou dans le modèle. Le premier niveau concerne le diagramme où les entités et les relations peuvent être floues. Le deuxième niveau concerne les occurrences des entités et des relations. Le troisième niveau concerne l'introduction du flou dans les valeurs des attributs.

D'autres extensions du modèle E-R ont été proposées (par Buckles et Petry, Ruspini et Vanenberghe notamment), que nous ne détaillons pas ici.

## 3.2 Bases de données orientées-objet floues

Le modèle de bases de données objet est apparu à la fin des années 80 et provient de l'introduction des caractéristiques des langages orientés objet aux bases de données [DEL 91, BEN 93]. Il permet la représentation de la base sous la forme d'une collection d'objets, où chaque objet est caractérisé par une structure et une interface. Les concepts fondamentaux liés à ce modèle sont détaillés ci-dessous.

- Objets *complexes* : ceux-ci sont construits (récursivement) sur des objets *atomiques* comme les entiers, réels, booléens, etc.
- Relations : un constructeur *n-uplet* permet de définir des relations unidirectionnelles ou bidirectionnelles entre objets.
- Identité d'un objet : chaque objet possède une identité propre indépendante de sa valeur (partage de sous-composants, cyclicité possibles).
- Encapsulation : un objet contient les méthodes qui permettent à un opérateur extérieur de le manipuler. Sa structure interne est cachée.
- Classes : groupes d'objets ayant une structure interne (attributs, méthodes) identique. Elles possèdent une description d'interface et d'implémentation.
- Héritage : manière de spécialiser une hiérarchie de classes, par l'ajout de nouveaux attributs, méthodes et/ou relations, avec une représentation par un graphe cyclique orienté.

Ces bases permettent la représentation de l'imperfection. Trois niveaux de flou sont souvent considérés [YAZ 92, MOU 95], décrits ci-dessous.

- Attributs : les valeurs des attributs tels l'âge peuvent être floues.
- Objets : un objet peut ne pas être entièrement une instance de sa classe. On considère alors un degré de certitude sur l'appartenance d'un objet à une classe, ce degré est associé à la relation *instance/classe*.

- Hiérarchie de classe : l'aspect le plus puissant des modèles de bases de données objet concerne la gestion des hiérarchies et des héritages. L'extension de ces hiérarchies au flou est donc un point majeur ; elle est réalisée en considérant un degré dans les relations du type *sous-classe/super-classe*.

## 3.3 Bases de données relationnelles floues

Le modèle relationnel est apparu dans les années 70, introduit par Codd [COD 70], [MAI 83, ULL 88]. L'introduction du flou dans ces bases a pris plusieurs formes, comme le souligne [BOS 98d] : représentation de données floues et appartenance graduelle des n-uplets aux relations, requêtes floues, dépendances fonctionnelles floues, et même sécurité.

### 3.3.1 Représentation des données

On rappelle que dans une base de données relationnelle, une relation $R$ sur un ensemble d'attributs $D_1, ..., D_n$ de domaines respectifs $dom_1, ..., dom_n$ est un sous-ensemble du produit cartésien de ces domaines.

Plusieurs moyens ont été étudiés pour traiter les imprécisions et les incertitudes dans les bases de données. Une étude des différentes sémantiques associées aux degrés introduits est menée dans [BON 95].

La plupart des modèles procèdent par l'extension des notions de domaines d'attribut et de relations (voir par exemple [BUC 93]). Par exemple, on considère une relation sur les attributs (domaines) *Nom, Age, Salaire*. Le n-uplet $(Dupont, 25, 2000; 0.8)$ traduit alors une imprécision liée au degré de certitude pour que le salaire de M. Dupont soit 2000. Le n-uplet $(Dupont, jeune, [2000, 2200])$, quant à lui, contient une valeur floue (*jeune*) et une valeur imprécise ([2000,2200]).

Comme l'indiquent [CHA 94, BOS 98a], il existe donc deux types d'imprécision.

Ainsi, le premier n-uplet précédemment considéré est associé à un poids (compris entre 0 et 1) qui représente soit le degré global de confiance dans les informations contenues dans le n-uplet, soit le degré d'appartenance de ce n-uplet à la relation.

On comprend mieux cette seconde interprétation en considérant un autre type de relation comme la relation $AIME$ sur les attributs $Nom$ et $Film$ d'une base de données cinématographique. Le poids peut alors indiquer le *degré de certitude* pour qu'une personne aime un film, ou alors indiquer avec *quel degré* une personne aime un film.

[BOS 97b] introduit trois types de bases de données floues :

- La première forme étend les relations aux relations floues. Chaque n-uplet défini sur le produit cartésien des domaines est alors associé à une valeur caractérisant le degré d'appartenance du n-uplet à la relation floue. Ce type de relation floue peut être utilisé pour stocker des données ou pour les requêtes. Les deux interprétations précédemment citées pour l'exemple de la relation $AIME$ sont possibles.
- La deuxième forme de bases de données floues proposée permet de prendre en compte les imprécisions du type de celles dues aux erreurs de mesure. On associe à chaque domaine d'attributs une relation de proximité (*interchangeabilité*). Cette relation indique à quel point deux valeurs d'un domaine sont proches.
  [BOS 97a] reprend cette approche et aborde le problème de la redondance dans les bases de données possibilistes en introduisant des méthodes de comparaison entre valeurs imprécises.

– La troisième forme de bases de données floues fournit le moyen de définir une distribution de possibilité à l'intérieur des n-uplets. Ainsi, si un des attributs de la relation n'est pas parfaitement connu, il pourra être exprimé selon une distribution de possibilité $\pi_A(x)$ prenant ses valeurs dans l'intervalle [0,1].

Dans [HAL 95], les relations floues sont utilisées dans le cadre de la découverte de connaissances et de la sécurité des bases de données pour appréhender les données communément traitées dans le monde réel, c'est-à-dire des données imprécises ; les n-uplets appartiennent alors aux relations avec des degrés entre 0 et 1.

Dans [UMA 93], le système FREEDOM (**F**uzzy **RE**lational **E**xtension for **D**ata **O**rganization and **M**anipulation) est présenté. Il s'agit d'une extension du modèle relationnel tel qu'il a été introduit par Codd. Les extensions sont dues aux modifications des définitions classiques.

Ainsi, une *relation étendue* est un sous-ensemble du produit cartésien d'une collection de distributions de possibilité. Une *base de données floues* est alors définie comme un ensemble de relations étendues. De telles bases permettent ainsi de gérer les données parfaitement connues, les données incertaines, et les données imprécises.

Un langage de manipulation est fourni, et une implantation logicielle en FORTRAN est proposée.

[MED 94] introduit un modèle de bases de données relationnelles floues. Dans ce modèle, l'information est organisée selon des *domaines* et des *relations* généralisés au flou qui sont des sous-ensembles du produit cartésien de domaines couplés à des *attributs de compatibilité* qui prennent leurs valeurs dans l'intervalle $[0, 1]$.

L'implantation logicielle de ce modèle est présentée dans [MED 95].

### 3.3.2 Dépendances fonctionnelles

Les dépendances fonctionnelles permettent de définir des schémas de bases robustes (pour les mises à jour) et d'éliminer les réplications. Elles ont, elles aussi, été étendues aux dépendances fonctionnelles floues [BOS 98a, BOS 97b, BOS 98b].

Dans le cadre classique, une dépendance fonctionnelle est définie de la manière suivante :

$(X \rightarrow Y \; est\, valide\, sur\, R) \Leftrightarrow (\forall t_1, t_2 \; t_1.X = t_2.X \Rightarrow t_1.Y = t_2.Y)$,

$X$ et $Y$ étant des ensembles d'attributs de la relation et $t_1.X$ la projection du n-uplet $t_1$ sur $X$.

Cette définition a été étendue aux dépendances fonctionnelles floues. On donne ici trois exemples.

1. [RAJ 88] définit la notion de dépendance fonctionnelle floue de la manière suivante :

$$(X \rightsquigarrow Y) \text{ est valide ssi}$$
$$(\forall t_1, t_2 \in R : \mu_{EQ}(t_1.X, t_2.X) \Rightarrow_{RG} \mu_{EQ}(t_1.Y, t_2.Y)$$

où

– $\Rightarrow_{RG}$ est l'implication de Rescher-Gaines avec

$$(a \Rightarrow_{RG} b) = \begin{cases} 1 & \text{si } a \leq b \\ 0 & \text{sinon} \end{cases}$$

- $\mu_{EQ}(a,b)$ est une mesure de similarité entre deux valeurs définies par leurs distributions de possibilité $\pi_a$ et $\pi_b$ sur l'univers $U$ avec $\mu_{EQ}(a,b) = \min_{u \in U}(1 - |\pi_a(u) - \pi_b(u)|)$.

2. [CUB 94] introduit la définition suivante :

   $(X \rightsquigarrow_{\alpha,\beta} Y) \Leftrightarrow (\forall t_1, t_2 \in R,\ \mu_{EQ_1}(t_1.X, t_2.X) \geq \alpha \Rightarrow \mu_{EQ_2}(t_1.Y, t_2.Y) \geq \beta)$
   où $\mu_{EQ_1}$ et $\mu_{EQ_2}$ sont des mesures de similarité.

3. [CHE 95] propose la forme suivante : $X \rightsquigarrow_{\varphi} Y$ est valide ssi
   $\forall t_1, t_2 \in R$, si $t_1.X = t_2.X$
   alors $t_1.Y = t_2.Y$
   sinon $(\mu_{EQ}(t_1.X, t_2.X) \Rightarrow_{GO} \mu_{EQ}(t_1.Y, t_2.Y)) \geq \varphi$
   où $\Rightarrow_{GO}$ est l'implication de Gödel définie par
   $$(a \Rightarrow_{GO} b) = \begin{cases} 1 & \text{si } a \leq b \\ b & \text{sinon} \end{cases}$$

Cependant, selon [BOS 98b], ces définitions ne sont pas vraiment satisfaisantes puisqu'elles n'ont pas la valeur sémantique attendue pour fournir des décompositions adéquates. Par exemple dans le cas de [CUB 94], une dépendance fonctionnelle $X \rightsquigarrow Y$ peut être valide alors qu'il existe des n-uplets ayant la même valeur sur les attributs $X$ mais pas sur les attributs $Y$. Une approche plus sémantique est donc proposée. Dans cette approche, on a :
$\forall t_1, t_2 \in R$ si $t_1.X = t_2.X$
alors $t_1.Y = t_2.Y$
sinon $\mu_{EQ}(t_1.X, t_2.X) \Rightarrow_{RG} \mu_{EQ}(t_1.Y, t_2.Y)$
où $\mu_{EQ}((a_1, ..., a_n), (b_1, ..., b_n))$
$= \sup_{u_1,...,u_n} \min(\pi_{a_1}(u_1), ..., \pi_{a_n}(u_n), \pi_{b_1}(u_1), ..., \pi_{b_n}(u_n))$

On peut ainsi comparer deux ensembles d'attributs $(a_1, ..., a_n)$ et $(b_1, ..., b_n)$ définis sur l'univers $U$.

D'autres extensions peuvent être envisagées, comme par exemple une action sur la force du quantificateur *pour tout* en le remplaçant par *pour la plupart* [BOS 97b].

### 3.3.3 Requêtes flexibles

Les requêtes flexibles ont été étudiées dans le cadre de bases classiques et dans le cas de bases contenant des informations imprécises ou incertaines. On entend par requête *flexible* une requête appliquée à des bases relationnelles étendues aux données imprécises, ou une requête dont les termes sont vagues [DUB 97a]. Par exemple, on interroge une base de données pour en extraire les individus d'âge *jeune* où *jeune* est un concept représenté par un sous-ensemble flou. La base peut être classique (non floue, par exemple [AND 95]) ou imprécise. Certains modèles représentent les résultats des requêtes dans le modèle initial, définissant ainsi des opérateurs de manipulation clos ([MED 94] par exemple), mais ce n'est pas toujours le cas. Les modèles qui ne représentent pas les résultats des requêtes souffrent donc de l'impossibilité de combiner des requêtes pour former des requêtes complexes, ou d'enchaîner des requêtes.

Dans [MED 94], on appelle *sélection atomique* une requête sur une relation $R_{FG}$ dans laquelle on estime la satisfaction d'une condition *simple*. Quand un attribut, un opérateur

et une constante flous sont impliqués dans une sélection simple, une telle condition sera satisfaite avec un degré variant entre 0 et 1. Un seuil de satisfaction peut alors être introduit, et tous les n-uplets ne satisfaisant pas la condition de manière suffisante sont éliminés. Le résultat d'une sélection atomique sur une relation est une relation (l'algèbre est close) et le degré de satisfaction de la condition est représenté dans l'attribut de compatibilité. Les degrés de cet attribut peuvent être combinés en cas de composition de requêtes et les n-uplets obtenus par chacune des requêtes simples sont également combinés (par l'union ou l'intersection selon que la combinaison de requêtes est de type disjonctive ou conjonctive).

[BOS 98d] distingue différents prédicats flous selon qu'ils sont *atomiques* (définis par une fonction d'appartenance - par exemple *grand*), *modifiés* (par un modificateur - par exemple *très* grand), ou *composés* (on a alors une combinaison de prédicats atomiques), et présente les deux types de requêtes flexibles existant : requêtes flexibles sur les bases non floues, et requêtes flexibles sur les bases floues.

De plus, les auteurs envisagent des requêtes mettant en jeu des quantificateurs, par exemple *la plupart* ou encore *une douzaine*. Cette idée est reprise dans [BOS 00b] qui étudie les requêtes du type "trouver les entreprises où le nombre d'employés *jeunes* est *très largement supérieur à 4*" ou "trouver les entreprises où le nombre d'employés *jeunes* est supérieur au nombre d'employés *bien payés*". Ici, *très largement supérieur à 4* est représenté par un sous-ensemble flou.

Pour traiter de telles requêtes, les auteurs étudient les définitions existantes de cardinalité d'ensembles flous : $\Sigma$-comptage pour les ensembles finis et discrets, les cardinalités définies par Dubois et Prade, ou encore les *sacs de degrés* de Yager. La première définition n'est pas envisagée car le $\Sigma$-comptage peut retourner la même cardinalité pour des ensembles très différents. La seconde est également rejetée. Les auteurs étudient donc plutôt la dernière proposition. Cependant, ils montrent que cette définition n'est pas non plus adaptée à la problématique. En effet, cette définition est booléenne alors que dans le cadre de requêtes flexibles, les auteurs souhaitent un résultat graduel indiquant à quel point une cardinalité est plus faible qu'une autre. En outre, cette définition n'offre pas une relation totale et dans certains cas, les cardinalités ne peuvent pas être comparées. Ils proposent donc une définition, fondée sur les implications étendues.

De même que [BOS 98d], [BOS 92] et [BOS 97b] passent en revue les différents systèmes de requêtes flexibles. Les auteurs étudient également les problématiques liées à l'indexation des bases pour les requêtes flexibles. [BOS 92, BOS 95b] présentent le langage de requête $SQLf$ qui est une extension de SQL autorisant l'introduction du flou dans les requêtes.

[BOS 00c] étudie les différents types de requêtes sur les bases de données contenant des valeurs imprécises représentées par des distributions de possibilité et leurs sémantiques associées. En effet, les auteurs considèrent que les requêtes appliquées à de telles bases peuvent concerner soit les valeurs prises par les attributs, soit les représentations de ces valeurs. Il s'agit alors d'interroger la base pour extraire, par exemple, un ensemble d'objets connus avec une certitude supérieure à un seuil fixé.

Dans [YAG 93, LAR 98, LAR 99], les auteurs proposent d'étendre les requêtes posées par l'utilisateur en utilisant la logique floue. Une couche logicielle est alors implantée entre l'interface utilisateur et la base de données. L'utilisateur est en mesure de spécifier un degré d'importance lié à chacun des critères de la requête.

### 3.3.4 Calcul d'agrégats dans les bases de données floues

La notion d'agrégat a elle aussi été étendue aux bases de données floues, notamment dans [RUN 89, RUN 92], ainsi que dans [DUB 90] et [BOS 02].

On rappelle que la notion d'agrégat en bases de données renvoie à des opérateurs du type *somme, maximum, minimum, moyenne.*

Dans [RUN 89, RUN 92], les données sont représentées à l'aide de distributions de possibilité. Les agrégats sont alors calculés en utilisant les fonctions de comptage et somme étendues aux données et critères de requêtes flous.

Les auteurs distinguent deux types d'agrégation : les agrégations scalaires et les agrégations mettant en jeu un partitionnement des données. Par exemple, on considère que le poste occupé par un employé peut être défini par une distribution de possibilité (par exemple { 1 / Assistant , 0.8 / Associé }). Les partitionnements interviennent alors pour des requêtes du type *"Quel est le salaire moyen des employés pour chaque sexe ?"* pour des partitions sur des données précises, ou *"Quel est le salaire maximal par poste ?"* pour des partitions sur des données imprécises.

Les opérateurs sont décrits selon qu'ils s'appliquent à des données non floues ou à des données possibilistes.

Dans le premier cas, les opérateurs sont décrits ci-dessous.

L'opérateur de **comptage** $f_{count}$ compte le nombre de n-uplets d'une relation.

Les opérateurs **de somme** ($f_{sum}$), **minimum** ($f_{min}$) et **maximum** ($f_{max}$) sont définis en utilisant le principe d'extension et l'arithmétique floue [KAU 91]. On a alors[6] :

$$f_\Phi((A)(r)) = \{u/y|((u = \min_{i=1}^{m} \mu_i(u_{ki})) \wedge (y = \Phi_{ki=k1}^{km} u_{ki}))(\forall ki = k1, ..., km \ : \ 1 \leq ki \leq n)\}$$

où $\Phi \in \{sum, max, min\}$, $r$ est la relation de la base à laquelle on s'intéresse, $A$ un attribut de la relation $r$ sur lequel porte l'agrégation, $m$ le nombre de valeurs à agréger avec chaque valeur définie par une distribution de possibilité de la forme $\{\mu_i(u_{ki})/u_{ki}|1 \leq ki \leq n\}$.

**Exemple :** On considère une relation *personne* définie sur les attributs *nom* et *poids*, avec par exemple 2 n-uplets et deux distributions de possibilité décrivant les poids de 2 personnes (voir figure 13).

La première personne *Camille* a un poids précis de 65, on a donc $\pi_{poids(Camille)} = \{1/65\}$ et *Pauline*, la seconde a un poids *léger* où *léger* est défini par la distribution suivante :

$\pi_{léger} = \{1/50, 0.9/55, 0.7/60\}$.

| Nom | Poids |
|---|---|
| Camille | 65 |
| Pauline | *léger* |

FIG. 13 – Relation possibiliste *personne*

On a donc :

$f_{max}((Poids)(personne))$

$= f_{max}(\{1/max(50, 65), 0.9/max(55, 65), 0.7/max(60, 65)\})$

$= f_{max}(\{1/65, 0.9/65, 0.7/65\})$

$= \{1/65\}$

[6] On adopte ici la notation suivante, décrivant le sous-ensemble flou $F$ de l'univers $U$ de fonction d'appartenance $\mu_F$ : $F = \{\mu_F(u_1)/u_1, \mu_F(u_2)/u_2, \ldots, \mu_F(u_n)/u_n\}$ avec $u_i \in U$ pour $1 \leq i \leq n$.

$f_{sum}((Poids)(personne))$
$= \{1/65\} \oplus \{1/50, 0.9/55, 0.7/60\}$
$= \{1/(65+50), 0.9/(65+55), 0.7/(65+60)\}$
$= \{1/115, 0.9/120, 0.7/125\}$

L'**opérateur de moyenne** $f_{avg}$ est défini par :

$$f_{avg1}((A)(r)) = f_{sum}((A)(r)) \oslash f_{count}((A)(r))$$

On a donc : $f_{avg1}((Poids)(personne))$
$= \{1/115, 0.9/120, 0.7/125\} \oslash 2$
$= \{1/57.5\ ,\ 0.9/60\ ,\ 0.7/62.5\ \}$

Un autre opérateur est proposé, utilisant le concept de *PEV* (*Possibilistic Expected Value*) introduit par Zemankova et Kandel. La valeur de PEV pour un attribut $A$ est égale à

$$PEV(A) = \frac{\sum_{i=1}^{n} \pi(u_i)u_i}{n}$$

Les auteurs proposent donc d'utiliser cette formule pour le calcul de la moyenne. On considère un schéma de relation $R(.., A, ..)$ avec $A$ défini sur le domaine $U = \{u_1, ..., u_n\}$. Soit $r$ consistant en des n-uplets $t_i$ avec $1 \leq i \leq m$, avec $t_i.A = \{\mu_i(u_1)/u_1, ..., \mu_i(u_n)/u_n\}$. On a alors :

$$f_{avg2}((A)(r)) = \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} \mu_i(u_j)u_j}{\sum_{i=1}^{m} \sum_{j=1}^{n} \mu_i(u_j)}$$

On a donc :
$f_{avg2}((Poids)(personne)) = \frac{(1\cdot65)+(1\cdot50+0.9\cdot55+0.7\cdot60)}{(1)+(1+0.9+0.7)}$
$= 57.36$

Dans le cas où la relation doit être partitionnée, les auteurs distinguent les cas où l'attribut servant à la partition est flou ou non. Dans le cas d'un attribut classique, l'évaluation consiste à partitionner la relation, puis à appliquer un des opérateurs décrits ci-dessus (selon la requête) sur chacune des partitions.

Dans le cas d'un attribut décrit par une distribution de possibilité, on introduit la notion de *partition de niveau* $\alpha$ et les agrégats sont calculés sur tous les niveaux $\alpha$ possibles pour chacun des blocs obtenus par le partitionnement. Sur un univers discret, les blocs sont donc construits en considérant chaque valeur possible du domaine et un n-uplet de la relation appartient à un bloc pour la partition de niveau $\alpha$ si son degré de possibilité pour la valeur du domaine considéré est supérieur à $\alpha$.

## 3.4 Vers un modèle de bases de données multidimensionnelles floues

Concernant les bases de données multidimensionnelles, deux modèles seulement ont traité du problème de l'introduction de l'imprécision et de l'incertitude.

### 3.4.1 Proposition de Feng et Dillon

[FEN 99] propose aux analystes une vision plus naturelle des données et des résumés des données de la base, en introduisant des variables linguistiques et des modificateurs linguistiques au niveau des agrégats. A partir d'un cube classique décrivant par exemple des ventes de produit, les auteurs proposent 3 niveaux :

1. niveau d'agrégation classique (par exemple "La moyenne des ventes est de 87000"),
2. résumé du niveau 1 en termes linguistiques (*"les ventes sont faibles"*),
3. agrégation du niveau 2 pour estimer les proportions grâce à des quantificateurs linguistiques (*la plupart des ventes sont faibles*)

Cependant, ce modèle n'est pas dédié à la représentation des données imprécises et/ou incertaines issues du monde réel, et ne permet pas l'application de requêtes flexibles aux bases multidimensionnelles. Il se contente en effet de fournir des outils de visualisation des données à différents niveaux de généralité. En outre, aucun algorithme d'automatisation de la production des niveaux de résumés n'est proposé.

### 3.4.2 Modèle de Pederson, Jensen et Dyreson

Les travaux de Pederson, Jensen et Dyreson proposent un modèle dédié aux données complexes et certains de leurs travaux abordent la question de l'imprécision dans les cubes [PED 99b, PED 99a, PED 00a, PED 00b]. Cependant la gestion de l'imprécision est plutôt liée au problème des données manquantes pour certains niveaux de hiérarchie. Par exemple, on connaît la famille de maladies dont un patient est atteint mais pas la maladie elle-même. Les problèmes d'agrégation de telles données sont présentés.

Les principales caractéristiques de ce modèle sont les suivantes :

- Schéma n-dimensionnel : $(\mathcal{F}, \mathcal{D})$ avec $\mathcal{F}$ type de fait et $\mathcal{D}$ un ensemble de types de dimension $\{Ti\}$.
- Hiérarchies définies grâce à un ensemble de catégories, couplé à une relation d'ordre partiel et deux éléments bornes inférieure et supérieure (on a donc un treillis).
- 3 types d'opérations d'agrégations : $\Sigma, \phi, c$ où $\Sigma$ désigne l'ensemble des fonctions d'agrégation applicables aux données pouvant être sommées, $\phi$ l'ensemble des fonctions applicables aux données sur lesquelles on peut calculer une moyenne et $c$ l'ensemble des fonctions applicables aux données ni sommables, ni moyennables.
- Prise en compte de l'incertitude : ajout de probabilités p :
  - sur l'ordre partiel sur les valeurs de dimension,
  - et sur l'appartenance d'un n-uplet à la relation cube.
- Algèbre proche du modèle relationnel (définition des 5 opérations de base et de l'agrégation)

Concernant les requêtes, les réponses sont données selon ce que contient la base pour le niveau auquel elle a été formulée. Les auteurs distinguent plusieurs types de réponses :

- *conservative :* seules les valeurs connues sont données, et pas de données imprécises,
- *libérale :* la réponse contient les données qui *pourraient* appartenir à la réponse,
- *pondérée :* la réponse contient les données qui *pourraient* appartenir à la réponse. Ces données sont pondérées selon le degré d'appartenance.

Selon les auteurs, neuf caractéristiques doivent être présentes dans un système de bases de données multidimensionnelles :

1. Hiérarchies de dimensions explicitement définies,
2. Symétrie de traitement mesures/dimensions,
3. Hiérarchies multiples (parents multiples) d'où la présence de plusieurs chemins d'agrégation possibles,
4. Agrégation *correcte* des données (pas de comptage de doublons, pas d'addition de données non additives etc),
5. Hiérarchies non strictes,
6. Prise en compte de relations 1-n (par exemple, un patient peut avoir plusieurs maladies),
7. Gestion des changements des données intervenant au cours du temps,
8. Gestion des incertitudes (La maladie est connue à 90%),
9. Possibilité d'enregistrement de données à différents niveaux de granularité.

Toutefois, ce modèle ne permet pas de représenter imprécision et incertitude à tous les niveaux où nous souhaitons l'intégrer. Dans ces travaux, les auteurs proposent en effet un modèle dédié aux données complexes et abordent la question de l'imprécision dans les cubes, en considérant des données potentiellement manquantes pour certains niveaux de hiérarchie. Cependant ce modèle ne gère pas les données imprécisément connues ni les requêtes dont les critères seraient vagues.

# Deuxième partie

# Bases de données multidimensionnelles floues

Comme nous l'avons vu dans la partie précédente, les modèles de bases de données multidimensionnelles permettent la construction de bases de données dédiées à l'analyse. Cette analyse peut se conduire à différents niveaux de granularité, sur de gros volumes de données représentés de manière agrégée.

D'autre part, les modèles de bases de données floues sont particulièrement intéressants pour la représentation des données imprécises et incertaines et la prise en compte de requêtes vagues. Cependant, aucun modèle de bases de données multidimensionnelles n'intègre la représentation de données imparfaites et la prise en compte de requêtes multidimensionnelles floues. Nous définissons donc dans cette partie un modèle de bases de données multidimensionnelles floues et les opérateurs de manipulation [LAU 01a].

Le modèle de représentation des connaissances est décrit dans le chapitre 1. Les opérateurs de manipulation des connaissances font l'objet des chapitres 2 et 3. Les requêtes complexes et le langage de manipulation sont étudiés dans le chapitre 4.

# Chapitre 1

# Représentation des connaissances

L'extension du modèle multidimensionnel au traitement de données imparfaitement connues nécessite la redéfinition des entités de base. Ces définitions sont détaillées tout au long de ce chapitre : éléments (section 1.1), hiérarchies (section 1.2), dimensions (section 1.3), et cubes flous (section 1.4).

Dans la suite, nous reprendrons toujours le même exemple, considérant une base multidimensionnelle classiquement rencontrée. On considère un cube dont les dimensions sont Produit, Mois, Ville et Ventes, avec une relation cube $Produit \times Mois \times Ville \rightarrow Ventes$. Une hiérarchie est définie sur la dimension Mois.

## 1.1 Éléments

Étant donné un ensemble de référence $X$, on note $\mathcal{F}(X)$ l'ensemble des sous-ensembles flous de $X$. Remarquons que les singletons de $X$ en sont des cas particuliers. On rappelle qu'un sous-ensemble flou $A$ de $X$ est défini par une fonction d'appartenance $f_A$ qui associe à chaque élément $x$ de $X$ le degré $f_A(x)$, compris entre 0 et 1, reflétant une gradualité dans son appartenance à A (voir annexe A).

**Définition 1 (Valeur)** *Une* valeur *$v$ sur un ensemble de référence $X$ appartient à $\mathcal{F}(X)$ et peut être :* **(i)** *soit une valeur non floue,* **(ii)** *soit une valeur imprécise représentée par sa fonction d'appartenance.*

**Définition 2 (Élément)** *Etant donné un ensemble de référence $X$, un* élément *$e$ est défini par le couple $(v, d) \in \mathcal{F}(X) \times [0, 1]$ où :*

- *$v$ est une* valeur*,*
- *$d \in [0, 1]$ est le* degré *de confiance associé à la valeur $v$.*

**Exemple 1** *On donne ici des exemples d'éléments, repris ultérieurement dans la figure 20.*

- *L'élément $(faible, 1)$ correspond à une donnée imprécise et certaine.* faible *est alors représenté par un sous-ensemble flou (voir figure 14).*
- *L'élément* $(82800, 0.8)$ *correspond à une donnée entachée d'incertitude. Par exemple, on sait avec une certitude* 0.8 *que les ventes pour le produit P1 dans la ville V1 au mois M1 se sont élevées à* 82800 *unités.*
- *L'élément $(faible, 0.8)$ correspond à une donnée imprécise et incertaine.*
  *Par exemple, on sait avec une certitude* 0.8 *que les ventes pour le produit P1 dans la ville V1 au mois M1 ont été* faibles*.*

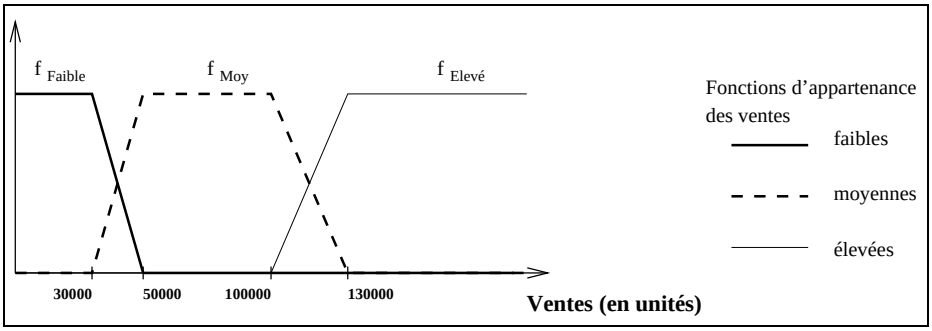


FIG. 14 – Ventes

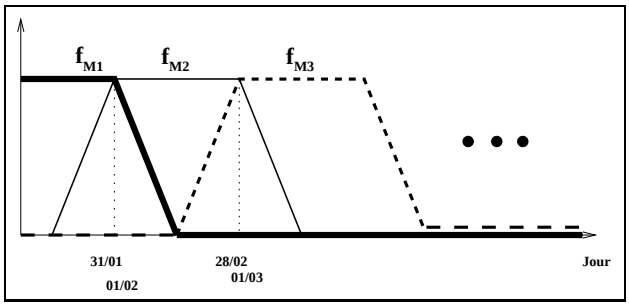


FIG. 15 – Sous-ensembles flous représentant les mois

## 1.2 Hiérarchies

Le modèle de cube repose sur l'analyse OLAP qui fait une place très importante aux opérations de drill-down et roll-up et la faculté de visualiser les données à des niveaux de granularité différents.

Comme le souligne [PED 99b], il est important de pouvoir définir plusieurs chemins de hiérarchie sur un même ensemble de valeurs, comme par exemple le cas classique décrit par la figure 18. Sur cette figure, les nœuds sont un sous-ensemble de valeurs dont la hiérarchie est définie soit par des partitions floues soit par des relations floues. Chaque nœud correspond à un niveau. Dans notre modèle, on appellera *hiérarchie simple* une hiérarchie de type arborescente et, à l'inverse, *hiérarchie multiple* les hiérarchies où plusieurs chemins peuvent être suivis d'un nœud fils vers les nœuds parents.

Cependant dans le cas flou, une hiérarchie simple peut contenir plusieurs chemins parents émanant d'un même nœud du fait des degrés de confiance qui leur sont affectés.

Cette section définit les différentes formes de hiérarchies qui coexistent dans notre modèle.

Les hiérarchies sont définies soit par des relations floues d'ordre strict (section 1.2.1), soit à l'aide de partitions floues ordonnées par une relation de finesse anti-symétrique $\prec$ (voir section 1.2.2). La figure 16 illustre ces deux types de hiérarchies. On note respectivement $H_P$ et $H_R$ les ensembles des hiérarchies des deux cas énoncés ci-dessus.

### 1.2.1 Hiérarchies reposant sur des relations floues

Les définitions d'une relation floue, et d'une relation floue d'ordre strict sont rappelées en annexe A [KLI 88].

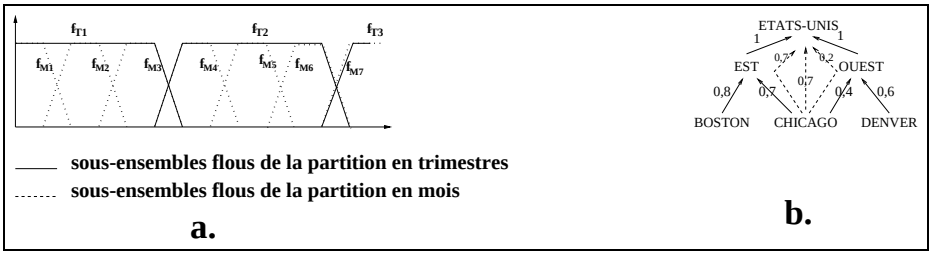


FIG. 16 – Hiérarchies floues

Dans notre cas, on se focalise sur les relations floues binaires sur un seul ensemble, ce qui correspond aux graphes orientés.

**Définition 3 (Hiérarchie simple incertaine)** *On dit qu'une hiérarchie définie sur $X$ est une* hiérarchie floue incertaine simple *si elle est définie par une relation floue d'ordre strict telle que pour tout $x \in X$, il existe au plus un $b \in X$ tel que $f_R(x, b) = 1$.*

**Définition 4 (Hiérarchie multiple incertaine)** *On dit qu'une hiérarchie définie sur $X$ est une* hiérarchie floue incertaine multiple *si elle est définie par une relation floue d'ordre strict telle qu' il existe $(x, y, z) \in X^3$ tel que $f_R(x, y) = f_R(x, z) = 1$.*

**Remarque :** Le cas non flou est le cas particulier où l'on considère un ensemble de valeurs associé à une relation définie sur une matrice ne contenant que des valeurs de l'ensemble {0,1} (matrice d'adjacence classique). Dans le cas flou, on pourra considérer les relations de niveau $\alpha$ ($\alpha \in [0, 1]$) pour agréger le long d'une dimension avec une certitude donnée.

### 1.2.2 Hiérarchies reposant sur des partitions floues

On définit également des hiérarchies par des partitions plus ou moins fines de l'univers. La finesse d'une partition par rapport à une autre et les hiérarchies de partitions sont détaillées ci-dessous.

La définition adoptée pour les partitions floues est la suivante :

**Définition 5 (Partition floue)** *Une partition floue d'un univers $X$ est une famille de sous-ensembles flous $\{F_1, \ldots, F_L\}$ de fonctions d'appartenance $f_1, \ldots, f_L$ telle que $\forall x \in X$,* $\sum_{i=1}^{L} f_i(x) = 1$.

**Remarque :** La définition suivante permet également de définir une partition floue par : $\forall x \in X$ : $\exists F_i$ *t.q.* $f_i(x) > 0$.

Cependant, on ajoute la contrainte de somme à 1 pour que, lors des opérations d'agrégation, la somme des pondérations à chaque endroit de l'univers soit égale à 1.

Les partitions sur lesquelles nous souhaitons travailler doivent représenter des hiérarchies intuitivement correctes. En effet, on pourrait considérer la définition de la finesse d'une partition, donnée par [BOU 95] :
$P_\alpha$ *plus fine que* $P_\beta \Leftrightarrow |P_\alpha| > |P_\beta|$ où $|P_\alpha|$ (resp. $|P_\beta|$) est le nombre de sous-ensembles flous constituant la famille de la partition $\alpha$ (resp. $\beta$).

Cependant, cette définition mène à des hiérarchies telles que celle décrite par la figure 17 (deux partitions sont définies sur l'univers, l'une en traits pleins et l'autre en traits pointillés) qui ne nous satisfait pas.

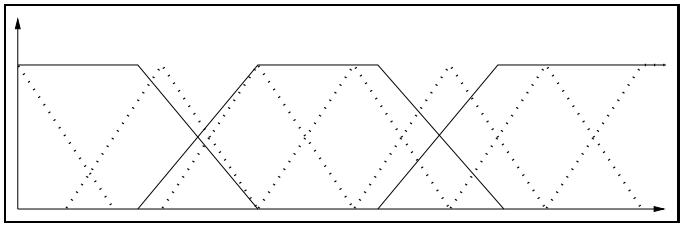

FIG. 17 – Hiérarchie floue contre-intuitive

Pour réaliser l'union des sous-ensembles flous, on considère la t-conorme de Lukasiewicz ($\perp(x,y) = min(x+y,1)$) afin de trouver des formes intuitivement correctes pour définir une hiérarchie.

On obtient alors des partitions et des hiérarchies du type de celles de la figure 16.a. Sur cette figure, on a $T_1 = \bigcup_{i=1}^{3} M_i$.

On définit donc la finesse de partitions floues de la manière suivante :

**Définition 6 (Finesse de partitions floues)** *On dit qu'une partition $P_\alpha = \{F_1, \ldots, F_{L_\alpha}\}$ est plus fine que la partition $P_\beta = \{F'_1, \ldots, F'_{L_\beta}\}$ dans la hiérarchie ($P_\alpha \prec P_\beta$) si*

1. *Pour tout $F' \in P_\beta$, il existe un intervalle $I \subset [1, L_\alpha]$ tel que $F' = \bigcup_{i \in I} F_i$*
2. *$P_\alpha \neq P_\beta$*

La relation $\prec$ est définie comme une relation antisymétrique sur les partitions. On peut montrer qu'elle est transitive, ce qui en fait une relation d'ordre strict (voir annexe C).

**Définition 7 (Successeur immédiat)** *Soit un ensemble de J partitions $\{P_i\}$ $(i = 1, \ldots, J)$ ordonnées par la relation de finesse $\prec$ antisymétrique. On dit que la partition $P_b$ est* successeur immédiat *de $P_a$ $(1 < a < J, 1 < b < J)$ si $P_a \prec P_b$ et il n'existe pas c $(1 < c < J)$ tel que $P_a \prec P_c \prec P_b$.*

**Définition 8 (Hiérarchie simple imprécise)** *On dit qu'une hiérarchie floue définie par la relation de finesse $\prec$ sur un ensemble de J partitions est une hiérarchie simple imprécise si toute partition $P_i$ $(i = 1, \ldots, J)$ a au plus un successeur immédiat.*

**Définition 9 (Hiérarchie multiple imprécise)** *On dit qu'une hiérarchie floue définie par la relation de finesse $\prec$ sur un ensemble de J partitions est une hiérarchie multiple imprécise s'il existe une partition $P_i$ $(i = 1, \ldots, J)$ ayant plus d'un successeur immédiat.*

## 1.3 Dimensions

Dans cette section, les entités de la base multidimensionnelle sont définies.

**Définition 10 (Dimension)** *Une dimension D est un quintuplet $\langle \nu, X, dom(D), H_P, H_R \rangle$ où :*

- *$\nu$ est le nom de la dimension,*
- *$X$ son univers de référence,*
- *$dom(D)$ un ensemble d'éléments nommé* domaine de la dimension *$D$,*
- *$H_P$ un ensemble de relations de finesse ($\prec$) sur des partitions définies sur les valeurs des éléments du domaine définissant des hiérarchies simples imprécises,*

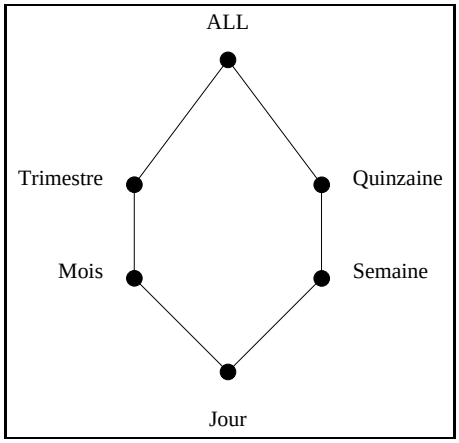


FIG. 18 – Hiérarchies multiples

– *$H_R$ un ensemble de relations floues d'ordre strict sur les valeurs des éléments du domaine définissant des hiérarchies simples incertaines.*

**Définition 11 (Valeurs du domaine)** *Soit $D$ une dimension. On appelle* valeurs du domaine *$dom(D)$ l'ensemble des valeurs $v$ telles que $(v, d)$ est élément de $dom(D)$. On note $Vdom(D)$ cet ensemble.*

Les hiérarchies ($h_P \in H_P$ et $h_R \in H_R$) doivent être *simples*; au besoin, plusieurs hiérarchies simples sont définies s'il existe sur le domaine une hiérarchie multiple. On évite ainsi les problèmes liés au choix du chemin de généralisation lors du passage entre différents niveaux de granularité.

**Exemple 2** *On donne trois exemples de dimension :*

1. ⟨Ventes,$\mathbb{R}^+$,{(faibles,1),(moyennes,1),(fortes,1)},∅,∅⟩ *est une dimension sans hiérarchie.*

2. *Soit P={J1,...,J15} la partition du domaine sur l'ensemble des jours, P1={S1,S2} la partition des 15 premiers jours en semaines, P2 la partition des premiers jours en mois, telles que $P \prec P1$ et $P \prec P2$.*
   *On définit h1 la hiérarchie des partitions P et P1 par la relation de finesse $\prec$ ($P \prec P1$). De même on définit h2 pour les partitions P et P2 ($P \prec P2$).*
   ⟨Temps,J,{(J1,1),...,(J15,1),(S1,1),(S2,1),(M1,1)},{h1,h2},∅⟩ *est une dimension temporelle hiérarchisée par partitions floues.*

3. *Soit R la relation floue d'ordre strict définie sur les valeurs V={ville1,ville2,ville3,Est,Ouest} par la matrice suivante :*

| *R* | *ville1* | *ville2* | *ville3* | *Est* | *Ouest* |
|---|---|---|---|---|---|
| *ville1* | *0* | *0* | *0* | *1* | *0* |
| *ville2* | *0* | *0* | *0* | *0.8* | *0.6* |
| *ville3* | *0* | *0* | *0* | *0* | *1* |
| *Est* | *0* | *0* | *0* | *0* | *0* |
| *Ouest* | *0* | *0* | *0* | *0* | *0* |

   ⟨Villes,V,{(ville1,1),(ville2,1),(ville3,1),(Est,1),(Ouest,1)},∅,{R}⟩ *est une dimension spatiale.*

**Définition 12** *On appelle* niveau le plus fin *d'une dimension hiérarchisée $D_i$ pour une hiérarchie donnée le sous-ensemble des valeurs des éléments de $dom(D)$ appartenant au niveau de granularité le plus fin pour cette hiérarchie.*

*On note $\underline{Vdom}(D)$ ce sous-ensemble.*

**Définition 13** *On appelle* niveau supérieur *d'une dimension hiérarchisée $D_i$ pour une hiérarchie donnée le sous-ensemble des valeurs des éléments de $dom(D)$ appartenant au niveau de granularité le plus grossier pour cette hiérarchie.*

*On note $\overline{Vdom}(D)$ ce sous-ensemble.*

Dans le cas de hiérarchies définies par des relations floues, si l'on considère la hiérarchie sur la dimension $D_i$ de domaine $dom(D_i)$ définie par la relation d'ordre floue R, on a :
$\underline{Vdom}(D_i) = \{a \in Vdom(D_i) | \nexists\, v \in Vdom(D_i)\ a \neq v \ \text{ et } \ f_R(v,a) > 0\}$
$\overline{Vdom}(D_i) = \{a \in Vdom(D_i) |\ \nexists v \in Vdom(D_i)\ a \neq v \ \text{ et } \ f_R(a,v) > 0\}$

Dans le cas de hiérarchies définies par des partitions, soit la dimension $D_i$ de domaine $dom(D_i)$, $\underline{Vdom}$ et $\overline{Vdom}$ on a :
$\underline{Vdom}(D_i) = \{a \in Vdom(D_i) |\ \nexists v \in Vdom(D_i)$ t.q. $a \neq v$ et $v \subseteq a\}$
$\overline{Vdom}(D_i) = \{a \in Vdom(D_i) |\ \nexists v \in Vdom(D_i)$ t.q. $a \neq v,\ a \subseteq v\}$

**Exemple 3** *On reprend l'exemple 2 :*

1. *Il n'y a pas de hiérarchie sur la dimension des Ventes donc*
   $\underline{Vdom} = \overline{Vdom} = Vdom = \{faibles, moyennes, fortes\}$
2. *Deux hiérarchies sont définies sur la dimension temporelle, et on a :*
   - $\underline{Vdom}_{h1} = \{J1, \ldots, J15\}$ *et* $\overline{Vdom}_{h1} = \{S1, S2\}$
   - $\underline{Vdom}_{h2} = \{J1, \ldots, J15\}$ *et* $\overline{Vdom}_{h2} = \{M1\}$
3. *Sur la dimension géographique, on a* $\underline{Vdom}_R = \{ville1, ville2, ville3\}$ *et* $\overline{Vdom}_R = \{Est, Ouest\}$

## 1.4 Cube flou

On note $\mathcal{D}$ un ensemble de dimensions. Un cube est alors défini comme une application du produit cartésien d'un ensemble de dimensions vers le produit cartésien de la dimension choisie comme mesure et de l'intervalle $[0,1]$. Par exemple, le cube $C$ de la figure 19.b est défini par :

$$C : PRODUIT \times MOIS \times VILLE \rightarrow VENTES \times [0,1]$$

**Définition 14 (Cube flou)** *Un cube flou est une application $C : D_1 \times \ldots \times D_k \rightarrow D_c \times [0,1]$ où $D_i \in \mathcal{D}$ $(i = 1, ..., k)$, et $D_c \in \mathcal{D}$ est la* mesure.

**Remarque** Une définition plus générale pourrait étendre la précédente en considérant un ensemble de mesures $\{D_{c_1}, ..., D_{c_m}\} \subset \mathcal{D}$. On a alors $C : D_1 \times \ldots \times D_k \rightarrow D_{c_1} \times \ldots \times D_{c_m} \times [0,1]$. Cependant, la partie droite est souvent simple et on retrouve le cas précédent : $C : dom_1 \times \ldots \times dom_k \rightarrow dom(c) \times [0,1]$.

La notion de *domaine actif*, définie dans le cadre des bases de données relationnelles ([MAI 83]), est reprise ici pour les domaines des dimensions.

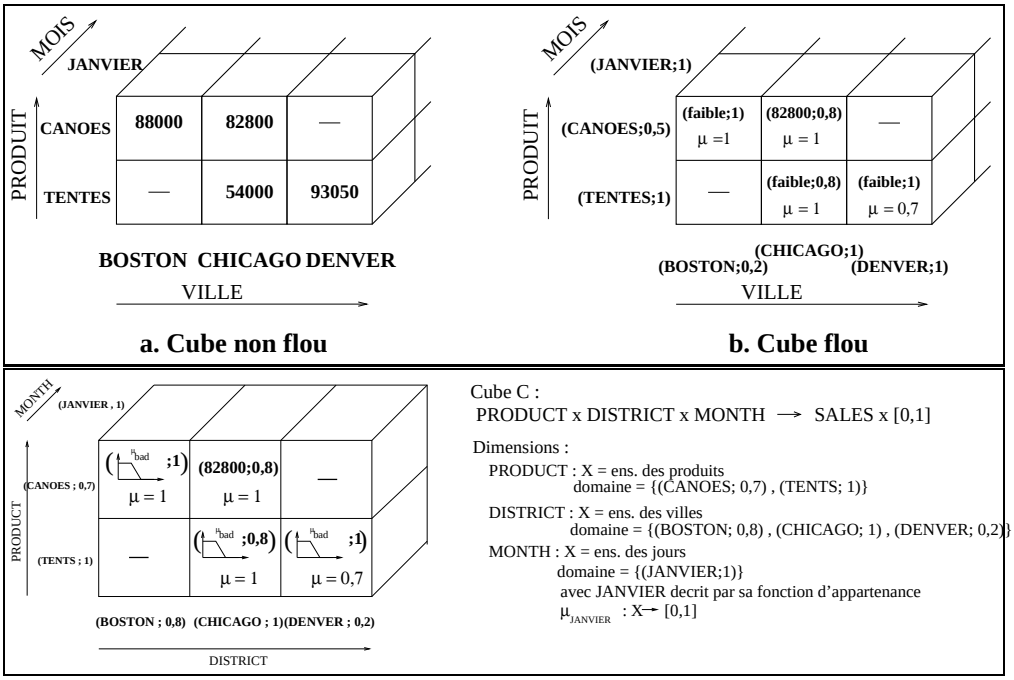


FIG. 19 – Exemples de cube

**Définition 15 (Domaine actif)** *Le domaine actif* $adom_C(D)$ *d'une dimension* $D$ *pour un cube* $C$ *donné est la partie de* $dom_C(D)$ *définie comme suit :*
$adom_C(D) =$
$\{e = (v, d) \in dom(D) \,|\, (d > 0) \wedge (\exists$ *une position de cellule* $\overrightarrow{x} = (\ldots, v, \ldots)$ *valide t.q.* $\mu(\overrightarrow{x}) > 0)\}$

*où* $\overrightarrow{x}$ *désigne une cellule par sa position le long des dimensions du cube.*

De même, la notion de valeurs du domaine actif est définie à partir du domaine actif :

**Définition 16 (Valeurs du domaine actif)** *On appelle* valeurs du domaine actif *d'une dimension* $D$ *l'ensemble des valeurs* $v$ *telles que* $(v, d)$ *est élément du domaine actif* $adom(D)$. *On note* $aVdom(D)$ *cet ensemble.*

**Exemple 4** *On reprend l'exemple précédent. Les éléments de la dimension* $Ventes$ *sont :* $(faibles, 1)$, $(moyennes, 1)$, $(fortes, 1)$ *où* $faibles$, $moyennes$ *et* $fortes$ *sont des sous-ensembles flous. Une fois utilisée dans le cube* $Produit \times Mois \times Ville \rightarrow Ventes$, *cette dimension devient la* mesure *du cube et les degrés associés aux différentes valeurs imprécises reflètent le degré de confiance sur la valeur.*

*La figure 20 illustre l'appartenance graduelle des cellules au cube : on sait par exemple que* (P1,V1,M1,faibles) *appartient au cube des ventes avec un degré* 0.7.

L'exemple suivant illustre la différence entre les différents degrés pour montrer l'importance et la signification du degré $\mu$.

**Exemple 5** *Imaginons que le cube ait été construit à partir de résultats envoyés par les magasins. Dans un premier cas, on a le courrier attestant que la vendeuse du magasin de la ville* v croit *se souvenir (avec une certitude 0.8) que ses ventes du produit* p *pour le mois* m *ont été faibles.*

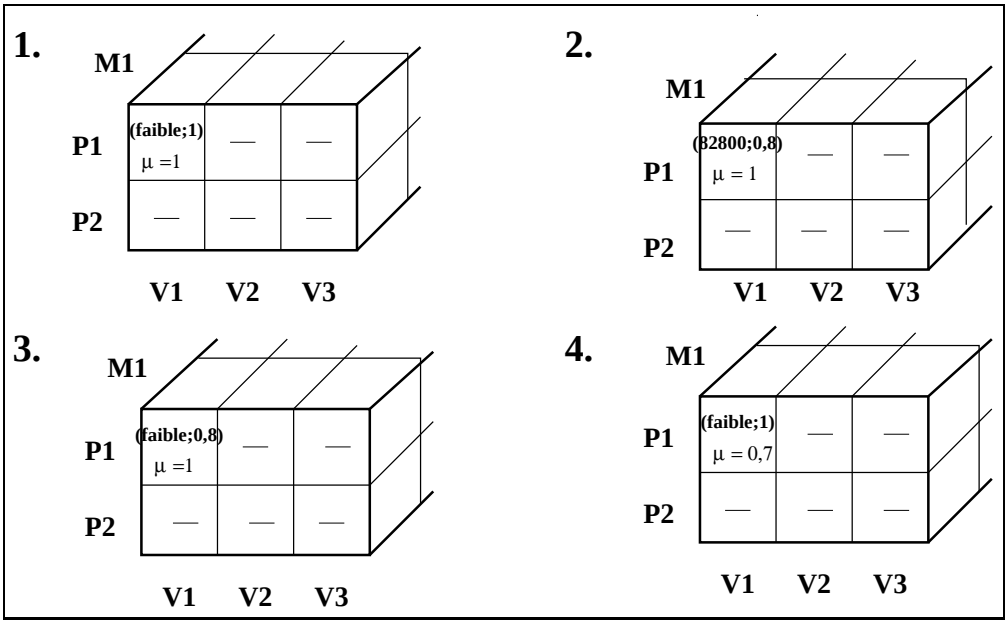


FIG. 20 – Exemples de cubes flous

*Dans un autre cas, on croit se souvenir (degré 0.2) que l'on a reçu un courrier mentionnant ces mêmes informations. On associe alors à la source d'information une certaine fiabilité*[7] *(qui dans ce cas vaudrait 0.2).*

*Dans le premier cas, la cellule à la position (p,m,v) sera (faible,0.8) et* $\mu_C(p,m,v) = 1$ *et dans l'autre on aura toujours E(C)(p,m,v)=(faible,0.8), qui peut également s'écrire* $v_C(p,m,v) = faible$ *et* $d_C(p,m,v) = 0.8$*, mais cette fois* $\mu_C(p,m,v) = 0.2$*.*

*On pourrait également envisager de reporter ce degré dans les différents degrés des différentes entités, et notamment dans le degré* $d_C$ *associé à la valeur de la cellule. Cependant ceci n'est pas équivalent à la solution que nous avons adoptée car le degré* $\mu$ *porte sur l'ensemble* $(p,m,v,E(C)(p,m,v))$ *et non pas seulement sur la valeur même de la mesure dans la cellule.*

**Exemple 6** *Considérons une base provenant de résultats donnés par des capteurs flous. Un cube de la base sera donc empli de données imprécises et/ou incertaines et tous les degrés* $\mu$ *seront à 1 (on est certain d'avoir obtenu ces valeurs). Les degrés* $\mu$ *seront alors importants lors de l'application d'une opération (de sélection par exemple) qui les modifiera pour exprimer le degré avec lequel chaque cellule a été sélectionnée ou non.*

## 1.5 Base de données multidimensionnelle floue

Une base de données multidimensionnelle floue est définie à partir des entités présentées dans ce chapitre :

**Définition 17** *Une base de données multidimensionnelle est un couple* $\langle \mathcal{D}, \Gamma \rangle$ *où :*

– $\mathcal{D}$ *est un ensemble de n dimensions* $D_i$ $(i = 1, \ldots, n)$ *sur les domaines* $dom(D_i)$*.*

[7] Cette approche est intéressante puisque la plupart des bases de données multidimensionnelles existantes sont construites à partir de diverses sources de données dont les fiabilités sont variables. Le problème de la gestion de la qualité des données est un point crucial dans les entrepôts de données et dans les systèmes décisionnels.

– $\Gamma$ *est un ensemble de cubes C*

## 1.6 Notations

On rappelle ici les notations utilisées pour représenter les différents éléments du cube ainsi que les hiérarchies.

- on note $\mathbf{E(C)}$ les éléments des cellules du cube C.
  $E(C)(d_1, \ldots, d_k)$ est l'élément du cube à la position $d_1$ sur la dimension $D_1$ ... $d_k$ sur la dimension $D_k$ ($d_i \in dom(D_i)$ $\forall i = 1, \ldots, k$).
  Afin de simplifier les notations, on note $\overrightarrow{x}$ le vecteur $(d_1, \ldots, d_k)$ et donc $E(C)(\overrightarrow{x})$ l'élément à la position $(d_1, \ldots, d_k)$.
- ces éléments sont constitués d'une valeur $\mathbf{v_C}(\overrightarrow{x})$ et d'un degré $\mathbf{d_C}(\overrightarrow{x}) \in [0, 1]$ qui vaut 1 dans le cas non flou.
- chaque élément de la relation cube $C$ est associé à un degré $\boldsymbol{\mu}_{\mathbf{C(d_1,\ldots,d_k)}}$.
- en ce qui concerne les éléments sur les dimensions, on note $\mathbf{v(d_i)}$ la valeur et $\mathbf{d(d_i)} \in [0, 1]$ le degré pour représenter les $\mathbf{d_i}$ de la dimension $D_i$ sur le domaine $dom(D_i)$.
- une hiérarchie de dimension sera représentée par :
  - un ensemble $\mathbf{H_P}$ de hiérarchies imprécises simples. Chaque hiérarchie $h \in H_P$ est définie par un ensemble de partitions ordonnées par la relation de finesse $\prec$. Une partition est une famille
    $\{F_1, \ldots, F_n\} \subseteq Vdom$ de $n$ sous-ensembles flous où $F_i$ est un sous-ensemble flou de fonction d'appartenance $f_i$,
  - ou un ensemble $\mathbf{H_R}$ de relations d'ordre floues R définies sur les valeurs des éléments du domaine de la dimension. On note M(R) la matrice associée[8] et $\mathbf{R_T}$ la fermeture transitive de R.

## 1.7 Discussion

Nous étudions ici de manière plus approfondie chacun des degrés introduits, afin d'en présenter la signification et la nécessité. Tous les degrés de notre modèle ont une sémantique qui leur est propre, et qui rend leur existence nécessaire à la représentation de données issues du monde réel. On distingue trois types de degrés :

- les degrés des *éléments* des cellules $\overrightarrow{x}$ ($d(\overrightarrow{x})$),
- les degrés sur les domaines de dimensions ($d(d_i)$ où $d_i$ est un élément du domaine actif de la dimension considérée),
- les degrés d'appartenance des cellules ($\mu(\overrightarrow{x})$).

Le premier type de degré $d(\overrightarrow{x})$ représente la confiance associée à la valeur correspondante $v(\overrightarrow{x})$, tandis que le second indique à quel point les tranches appartiennent au cube. Le troisième degré est associé à l'information complète donnée par la position de la cellule (éléments des domaines de toutes les dimensions) et l'*élément* $(v(\overrightarrow{x}), d(\overrightarrow{x}))$ lui-même. Il indique à quel point l'information complète appartient au cube.

Si l'on se réfère au cas relationnel, le premier degré indique une confiance sur une valeur d'attribut, tandis que le troisième indique à quel point le n-uplet appartient à la relation

[8] le domaine est un ensemble *fini* d'éléments.

considérée. Le deuxième type de degré, correspondant à l'appartenance d'une tranche au cube, est quant à lui très fortement lié au modèle multidimensionnel. Dans le modèle relationnel, il faudrait considérer une requête de type *group-by*. Ce degré indiquerait alors à quel point chacun des ensembles de n-uplets de la partition réalisée appartient à la relation.

Dans le cube *VENTES* de l'exemple précédent, les degrés $\mu$ sont associés à l'information complète donnée par la *ville*, le *produit*, le *mois*, et la valeur des *ventes*, et non pas seulement à la valeur des *ventes*. D'autre part, les degrés $d(\overrightarrow{x})$ ne sont associés qu'à la valeur $v(\overrightarrow{x})$ correspondante représentant le taux de ventes.

Ces degrés peuvent provenir des données elles-mêmes, par exemple pour un cube construit à partir de sources diverses. Dans ce cas, les degrés $\mu$ offrent la possibilité de prendre en compte les degrés de fiabilité des différentes sources de données. De plus, l'incertitude sur les valeurs elles-mêmes peut être prise en compte par les degrés $d(\overrightarrow{x})$.

Cependant, les degrés peuvent également provenir d'opérations successives de requêtes sur la base de données, comme nous le verrons dans le chapitre 2. Ces degrés indiquent à quel point les entités (cellules et tranches du cube) appartiennent au cube résultant d'une opération.

Le modèle proposé permet la représentation de connaissances imparfaites dans les bases de données multidimensionnelles. Nous introduisons maintenant les opérateurs de manipulation de telles bases.

Les opérateurs unaires sont présentés dans le chapitre 2. Ils sont distingués des opérateurs binaires introduits dans le chapitre 3.

Enfin, le chapitre 4 introduit un langage de manipulation ainsi que l'étude de la combinaison d'opérateurs.

Ces propositions ont fait l'objet de publications dans [LAU 01a, LAU 02b].

# Chapitre 2

# Manipulation des données : opérateurs unaires

Les utilisateurs de systèmes décisionnels visent essentiellement à visualiser les données sous formes de cubes et de sous-cubes par l'application d'opérations successives. Si les opérations de visualisation *pure* ne modifiant pas le contenu du cube (par exemple rotation, inversion de valeurs de dimension) ne sont pas altérées par l'introduction du flou, les opérations de sélection et de navigation à travers les niveaux de granularité nécessitent une redéfinition. Ce chapitre détaille les opérateurs de sélection et de généralisation.

## 2.1 Sélection sur les valeurs des cellules

L'opération *dice* consiste à sélectionner les cellules du cube qui satisfont un critère. Dans le cas classique non flou décrit par [AGR 97], en notant $C$ le cube sur lequel porte la sélection, chaque cellule appartient alors ou non au cube $C'$ résultant de l'opération, selon que le critère p est vérifié ou non :

$$E(C')(d_1,\ldots,d_k) = \begin{cases} E(C)(d_1,\ldots,d_k) & \text{si le critère p est vérifié} \\ \emptyset & \text{sinon} \end{cases}$$

Dans le cas de cubes flous, on associe non plus un degré booléen mais un degré appartenant à l'intervalle [0,1] indiquant à quel point la cellule appartient au cube résultat. Le degré reflète non seulement l'ancien degré d'appartenance de la cellule au cube de départ mais également le niveau de satisfaction du critère de restriction. Ce calcul nécessite l'utilisation de mesures de comparaison [BOU 96], le résultat étant reporté sur le degré $\mu$ d'appartenance de la cellule à la relation cube.

On note $C'$ le cube obtenu après une opération de sélection sur le cube $C$ en considérant le critère p représenté par un opérateur flou de fonction d'appartenance $\mu_O$. Les éléments des cellules du cube $C'$ sont les mêmes que dans le cube $C$.

On considère un opérateur d'agrégation $\mathcal{T}$, défini comme $\mathcal{T} : [0,1]\times[0,1]\times[0,1] \rightarrow [0,1]$. On pourra par exemple utiliser une t-norme $\top$ avec $\mathcal{T}(x,y,z) = \top(x,\top(y,z))$. Le choix de cet opérateur est détaillé dans la section 2.3. Pour les exemples, la t-norme min est choisie[9]. Selon une première approche, les degrés $\mu$ d'appartenance des cellules au cube $C'$

[9]Pour simplifier les notations, la fonction min étant d'arité quelconque, elle sera utilisée sans distinction avec 2 arguments ou plus.

sont calculés selon la formule suivante :

$$\boldsymbol{\mu}_{\mathbf{C}'}(\overrightarrow{x}) = \mathcal{T}(\mathcal{C}(\mu_{\mathbf{O}}, \mathbf{v}_{\mathbf{C}}(\overrightarrow{x})), \mathbf{d}_{\mathbf{C}}(\overrightarrow{x}), \mu_{\mathbf{C}}(\overrightarrow{x})) \tag{2.1}$$

où $\mathcal{C}$ est une mesure de satisfiabilité (voir annexe A).

Les degrés $d(d_i)$ sur les tranches sont inchangés et reportés sur le nouveau cube puisque les domaines restent les mêmes. Leur intégration dans ce calcul est donc inutile.

**Cas particuliers :**

L'expression (2.1) se simplifie si l'on considère les différents cas particuliers possibles. Pour ce faire, on considère la t-norme min pour définir l'opérateur $\mathcal{T}$. On rappelle les points suivants.

- Pour les cas précis, la valeur $v_C(\overrightarrow{x})$ est réduite à un singleton et on note $v_C(\overrightarrow{x}) = \{\dot{v}_C(\overrightarrow{x})\}$. [MAR 98b] montre que quand $v_C(\overrightarrow{x})$ est précise et que l'on utilise pour mesure de satisfiabilité la mesure (A.3) donnée en annexe A, alors on a :

$$\mathcal{C}(\mu_O, v_C(\overrightarrow{x})) = \mu_O(\dot{v}_C(\overrightarrow{x})) \tag{2.2}$$

- Pour les cas certains, on rappelle que 1 est élément neutre des t-normes.

Tous les cas particuliers possibles sont détaillés ci-dessous. On considère le critère de sélection $\mathcal{X}_O$ représenté par sa fonction d'appartenance $\mu_O$. Des exemples pour chacun de ces cas sont donnés ensuite.

1. *Valeurs non floues - opérateur flou*
   Dans ce cas, on a $d_C(\overrightarrow{x}) = 1$ puisque les valeurs sont certaines et $\mathcal{C}(\mathcal{X}_O, v_C(\overrightarrow{x})) = \mu_O(\dot{v}_C(\overrightarrow{x}))$ puisque les valeurs sont précises d'où :
   $\mu_{C'}(\overrightarrow{x}) = \top(\mu_O(\dot{v}_C(\overrightarrow{x})), \mu_C(\overrightarrow{x}))$
2. *Valeurs floues - opérateur non flou*
   - Valeurs précises incertaines
     Dans ce cas, on a $\mathcal{C}(\mathcal{X}_O, v_C(\overrightarrow{x})) = \mu_O(\dot{v}_C(\overrightarrow{x}))$ puisque les valeurs sont précises, d'où :
     $E(C')(\overrightarrow{x}) = E(C)(\overrightarrow{x})$ et $\mu_{C'}(\overrightarrow{x}) = \mathcal{T}(\mu_O(\dot{v}_C(\overrightarrow{x})), d_C(\overrightarrow{x}), \mu_C(\overrightarrow{x}))$
   - Valeurs imprécises, certaines
     Dans ce cas, on a $d_C(\overrightarrow{x}) = 1$ puisque les valeurs sont certaines, d'où :
     $$\begin{aligned} \mu_{C'}(d_1, \ldots, d_k) &= \mathcal{T}(\mathcal{C}(\mathcal{X}_O, v_C(\overrightarrow{x})), 1, \mu_C(\overrightarrow{x})) \\ &= \top(\mathcal{C}(\mathcal{X}_O, v_C(\overrightarrow{x})), \mu_C(\overrightarrow{x})) \end{aligned}$$
   - Valeurs imprécises et incertaines
     On a $v_C(\overrightarrow{x})$ valeur imprécise et $E(C)(\overrightarrow{x})$ entachée d'un degré d'incertitude $d_C(\overrightarrow{x})$. On calcule $\mu_{C'}(\overrightarrow{x})$ :
     $\mu_{C'}(\overrightarrow{x}) = \mathcal{T}(\mathcal{C}(\mathcal{X}_O, v_C(\overrightarrow{x})), d_C(\overrightarrow{x}), \mu_C(\overrightarrow{x}))$
3. *Valeurs floues - opérateur flou*
   On utilise une mesure de comparaison pour comparer les deux sous-ensembles flous. On peut reprendre les différents cas cités précédemment :

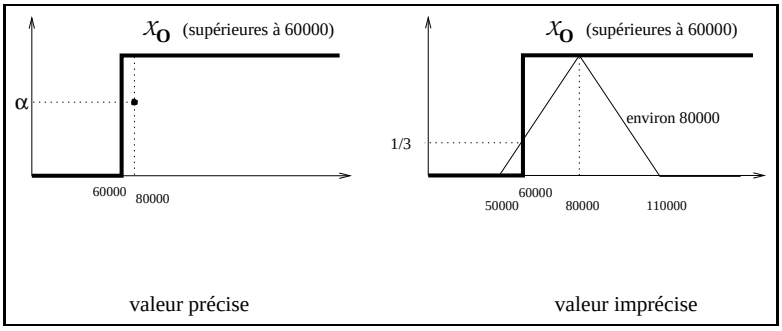


FIG. 21 – Valeurs floues, opérateur non flou

(a) Valeurs précises incertaines
La valeur est précise donc $\mathcal{C}(\mathcal{X}_O, v_C(\overrightarrow{x})) = \mu_O(v_C(\overrightarrow{x}))$
$\mu_{C'}(d_1, \ldots, d_k) = \mathcal{T}(\mu_O(v_C(\overrightarrow{x})), d_C(\overrightarrow{x}), \mu_C(\overrightarrow{x}))$

(b) Valeurs imprécises, certaines
Dans ce cas $d_C(\overrightarrow{x}) = 1$ d'où :
$\mu_{C'}(d_1, \ldots, d_k) = \top(\mathcal{C}(\mathcal{X}_O, v_C(\overrightarrow{x})), \mu_C(\overrightarrow{x}))$

(c) Valeurs imprécises et incertaines
Ce cas est le cas général :
On calcule $\mu_{C'}(\overrightarrow{x})$ en fonction de $v_C(\overrightarrow{x}), d_C(\overrightarrow{x})$ et $\mu_C(\overrightarrow{x})$ à partir de l'équation (2.1).

On illustre les différents cas particuliers détaillés ci-dessus par des exemples.

**Exemple 7** Critère non flou - valeurs précises incertaines.

*On considère l'exemple donné par la figure 21. On connaît le montant des ventes (80 000) pour une cellule à la position (p,m,v) avec un degré de certitude* $\alpha$*, et le critère de sélection est* $\mathcal{X}_O$ = Ventes supérieures à 60 000. *Cette cellule appartient au cube C avec un degré* $\mu(p, m, v) = \nu$.

*Cette même cellule appartiendra donc au cube résultat C' avec un degré* $\mu_{C'}(p, m, v)$ *valant :* $\min(\mu_O(80\ 000), \alpha, \nu) = \min(1, \alpha, \nu) = \min(\alpha, \nu)$.

**Exemple 8** Critère non flou - valeurs imprécises, certaines

*On considère une fois encore l'exemple donné par la figure 21. On connaît le montant des ventes (environ 80 000) pour une cellule avec certitude (degré* $= 1$*), et le critère de sélection est* $\mathcal{X}_O$ = Ventes supérieures à 60 000. *Cette cellule appartient au cube C avec un degré* $\mu(p, m, v) = \nu$.

*Cette même cellule appartiendra donc au cube résultat C' avec un degré* $\mu_{C'}(p, m, v)$ *valant :*

$\min(\mathcal{C}(\mathcal{X}_O, v_C(\overrightarrow{x})), \nu) \quad = \min(\mathcal{C}(\mathcal{X}_O, environ\,80000), \nu)$

– *En considérant la mesure de comparaison donnée par l'équation (A.1), on a :*
$\mathcal{C}(\mathcal{X}_O, v_C(\overrightarrow{x})) = \min(1 - 1/3, \nu)$.
*En effet, la droite passant par les points* $(50000, 0)$ *et* $(80000, 1)$ *coupe la droite* $x = 60000$ *en* $y = 1/3$. *D'où* $\mu_{C'}(p, m, v) = \min(2/3, \nu)$.

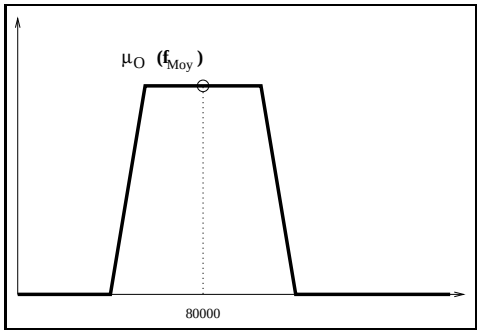


FIG. 22 – Valeurs non floues, opérateur flou

– *en considérant la mesure de comparaison donnée par l'équation (A.3), on a :*
$\mathcal{C}(\mathcal{X}_O, v_C(\overrightarrow{x})) = \frac{1/3\cdot(80000-60000)+(20000\cdot 2/3)/2}{(110000-50000)/2} = 4/9$. *D'où* $\mu_{C'}(p,m,v) = \min(4/9, \nu)$.

**Exemple 9** Critère non flou - valeurs imprécises et incertaines.

*On considère une cellule du cube C à la position (p,m,v) contenant la valeur imprécise (environ* 80000*) connue avec un degré de certitude* 0.5. *Le critère de sélection est* $\mathcal{X}_O =$ Ventes supérieures à 60000. *Cette cellule appartient au cube C avec un degré* $\mu(p,m,v) =$ 0.8. *On a alors :*

$$\mu_{C'}(p,m,v) \quad = \min(\mathcal{C}(\mathcal{X}_O, v_C(\overrightarrow{x})), d_C(\overrightarrow{x}), \mu_C(\overrightarrow{x}))$$

*soit*

$$\mu_{C'}(p,m,v) = \begin{cases} \min(2/3, 0.5, 0.8) = 0.5 & \textit{en utilisant la mesure (A.1)} \\ \min(4/9, 0.5, 0.8) = 4/9 & \textit{en utilisant la mesure (A.3)} \end{cases}$$

**Exemple 10** Critère flou - Valeurs non floues

*Le premier schéma de la figure 22 illustre cet exemple. La valeur de la cellule à la position (p,m,v) est connue de manière précise et certaine (cette valeur est 80 000). Cette cellule appartient au cube C avec un degré* $\mu_C(p,m,v) = \nu$. *On sélectionne les ventes moyennes.*

*On a donc :*
*v(C')(p,m,v)=80 000 et* $\mu_{C'}(p,m,v) = \min(\mu_O(v_C(p,m,v)), \mu_C(p,m,v))$.
*Donc :*

$\mu_{C'}(p,m,v) = \min(1, \nu) = \nu$

**Exemple 11** Critère flou - valeurs précises incertaines

*On considère dans ce cas le premier exemple de la figure 23. La valeur de la cellule à la position (p,m,v) est connue de manière précise (80 000) mais avec un degré de certitude valant* $\alpha$. *Cette cellule appartient au cube C avec un degré* $\mu_C(p,m,v) = \nu$.

*On a donc :*

$$\begin{aligned} \mu_{C'}(p,m,v) &= \min(\mu_O(80000), \alpha, \nu) \\ &= \min(1, \alpha, \nu) \\ &= \min(\alpha, \nu) \end{aligned}$$

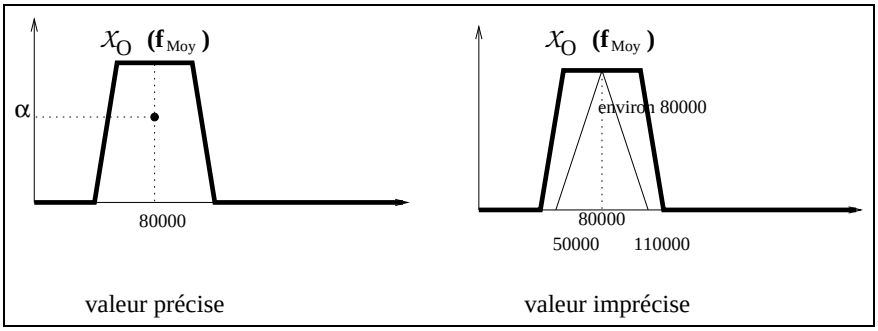


FIG. 23 – Valeurs floues, opérateur flou

**Exemple 12** Critère flou - valeurs imprécises, certaines

*On considère le deuxième exemple de la figure 23. La valeur de la cellule à la position (p,m,v) est connue de manière imprécise ($v_C(p,m,v)$ = environ 80 000) et de manière certaine ($d_C(p,m,v) = 1$). Cette cellule appartient au cube C avec un degré $\mu_C(p,m,v) = \nu$. Dans cet exemple, $\mathcal{C}(\mu_O, v_C(\overrightarrow{x}))$ vaut 1 car $v_C(\overrightarrow{x}) \subset \mu_O$.*

*On a donc :*

$$\begin{aligned}\mu_{C'}(p,m,v) &= \min(\mathcal{C}(\mathcal{X}_O, v_C(\overrightarrow{x})), \nu)\\ &= \min(1, \nu)\\ &= \nu\end{aligned}$$

**Exemple 13** Critère flou - valeurs imprécises et incertaines.

*On considère dans ce cas le deuxième exemple de la figure 23. La valeur de la cellule à la position (p,m,v) est connue de manière imprécise ($v_C(p,m,v)$ = environ 80 000), avec un degré de certitude $d_C(p,m,v)$. Cette cellule appartient au cube C avec un degré $\mu_C(p,m,v) = \nu$. On a $\mathcal{C}(\mathcal{X}_O, v_C(\overrightarrow{x})) = 1$.*

*On a donc :*

$$\begin{aligned}\mu_{C'}(p,m,v) &= \min(\mathcal{C}(\mathcal{X}_O, v_C(\overrightarrow{x})), d_C(\overrightarrow{x}), \nu)\\ &= \min(1, d_C, \nu)\\ &= \min(d_C, \nu)\end{aligned}$$

## 2.2 Sélection sur les valeurs des dimensions

Cette opération consiste à réduire le domaine actif de la dimension du cube sur lequel porte la sélection. On exprime alors $adom_{C'}(D_i)$ en fonction de $adom_C(D_i)$ et du critère de sélection p (flou ou non), où $C$ est le cube sur lequel porte la sélection et $C'$ le cube résultant de l'opération. Une sélection sur la dimension $D_i$ entraîne donc la suppression des tranches du cube correspondant aux valeurs $d_i$ qui ne satisfont pas le critère de sélection.

Si $C$ est la relation suivante $C : D_1 \times \ldots \times D_k \rightarrow D_C$, on a :

$\forall\, i = 1, \ldots, k$, $adom_{C'}(D_i) \subseteq adom_C(D_i)$

Dans le cas flou, on ne supprime pas la tranche, mais on modifie les valeurs d'appartenance des valeurs à la dimension. C'est donc le domaine actif des dimensions qui est modifié.

De même que pour les cellules dont le degré d'appartenance est 0, on supprime les valeurs $d_i$ de domaine de dimensions si $d(d_i) = 0$.

**Cas général.**

On note $adom_C(D_i) = \{(v_C(d_i), d_C(d_i))\}$ et $adom_{C'}(D_i) = \{(v_{C'}(d_i), d_{C'}(d_i))\}$.
Le cas général pour le calcul du cube résultat est donné par l'équation suivante :

$$\forall \mathbf{d_i} \in \mathbf{adom(D_i)} \ : \ \mathbf{d_{C'}(d_i)}{=}\theta(\mathcal{C}(\boldsymbol{\mu}_{\mathbf{O}}, \mathbf{v_C(d_i)}), \mathbf{d_C(d_i)}) \tag{2.3}$$

où $\theta$ est un opérateur, par exemple une t-norme $\top$, et $D_i$ est la dimension sur laquelle porte la sélection.

On détaille ci-dessous le résultat d'une telle opération de sélection dans les différents cas particuliers.

**Cas particuliers**

1. *Éléments non flous*
   Dans le cas où $d_C(d_i)$ égale 1, et où $v_C(d_i)$ est un nombre non flou, on a $\mathcal{C}(\mathcal{X}_O, v_C(d_i)) = \mu_O(\dot{v}_C(d_i))$ d'où :
   $d(d_i) = \mu_O(\dot{v}_C(d_i))$
2. *Valeurs floues*
   (a) Valeur précise $v_C(d_i) = \{\dot{v}_C(d_i)\}$ et $d_C(d_i)) \neq 1$. $\mathcal{C}(\mathcal{X}_O, v_C(d_i)) = \mu_O(\dot{v}_C(d_i))$ car la valeur est précise d'où :
   $d_{C'}(d_i) = \top(\mu_O(\dot{v}_C(d_i)), d_C(d_i))$
   (b) élément imprécis certain :
   On a $d_C(d_i) = 1$ d'où :
   $d_{C'}(d_i) = \mathcal{C}(\mathcal{X}_O, v(d_i))$
   (c) élément imprécis et incertains ($d_C(d_i) \neq 1$) :
   Dans ce cas, on retrouve l'expression du cas général décrit par l'équation (2.3).

Dans tous les cas, $dom_i(C') = \{d_i \in dom_i(C) \,|\, d_{C'}(d_i) \neq 0\}$. La figure 24 donne un exemple de cette opération.

## 2.3 Étude des opérateurs utilisés

Les deux opérations de sélection nécessitent la définition d'un opérateur d'agrégation des différentes informations, ainsi que la définition d'une mesure de comparaison entre des valeurs et un critère. Le choix des opérateurs d'agrégation nécessite la plus grande attention car il détermine la sémantique associée aux opérations proposées [KEL 96, DET 00]. Nous étudions donc ici les solutions proposées dans notre modèle sous l'angle à fois de la sémantique et des performances de calcul.

### 2.3.1 Comparaison entre valeur et critère

La mesure $\mathcal{C}$ est utilisée pour comparer les valeurs aux critères de sélection. Les mesures de comparaison, et parmi elles les mesures de *satisfiabilité* sont appropriées à un tel besoin, afin de mesurer à quel point une valeur $V$ répond à un critère $REF$ [BOU 96]. Nous choisissons la mesure suivante :

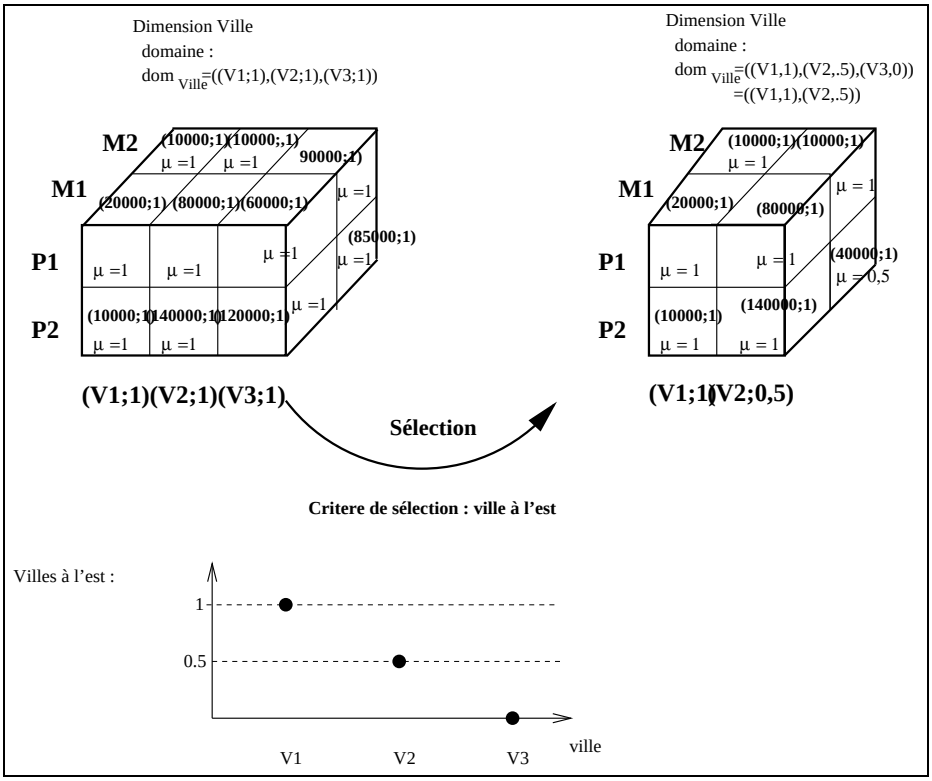


FIG. 24 – Sélection sur les valeurs d'une dimension

$$\mathcal{C}(V, REF) = \begin{cases} \frac{M(V \cap REF)}{M(V)} & \text{si } M(V) \neq 0 \\ 0 & \text{sinon} \end{cases}$$

où pour tout sous-ensemble flou $A$ défini sur $\Omega$, on a :

$$M(A) = \begin{cases} \sum\limits_{x \in \Omega} f_A(x) & \text{si } \Omega \text{ est dénombrable} \\ \int_\Omega f_A(x)dx & \text{sinon} \end{cases}$$

Cette mesure est choisie car elle permet de retrouver le cas classique si la valeur est connue de manière précise, c'est-à-dire si elle est représentée par un point, le sous-ensemble flou étant alors réduit à un singleton. Cette propriété est très intéressante afin d'augmenter les performances dans le cas de valeurs précises, ceci rendant notre système totalement équivalent aux systèmes classiques. La preuve de cette équivalence est donnée ci-dessous.

On note $\mu_{REF}$ et $\mu_V$ les fonctions d'appartenance des sous-ensembles flous décrivant respectivement le critère $REF$ et la valeur $V$. Si la valeur est précise, alors le sous-ensemble flou $V$ est réduit à un singleton $x_O$ tel que pour tout $x \neq x_O$, on a $\mu_V(x) = 0$ et $\mu_V(x_0) = 1$. On a donc $M(V \cap REF) = \mu_{REF}(x_0)$ et $M(V) = 1$. Le calcul est donc réduit au calcul du degré d'appartenance de la valeur $x_O$ au sous-ensemble flou représentant le critère : $C(V, REF) = \mu_{REF}(x_O)$ (voir [MAR 98b] pour le détail de la preuve).

**Remarque.** La comparaison de la valeur avec le critère de sélection pourrait s'effectuer en interprétant les sous-ensembles flous comme des distributions de possibilité et en utilisant les deux indicateurs de possibilité et de nécessité. En effet, l'utilisation d'une mesure de

satisfiabilité conduit à des résultats différents de 0 alors qu'il peut exister au moins une valeur de la distribution qui ne soit pas incluse dans la description du critère.

Plusieurs interprétations sémantiques sont possibles à partir du modèle proposé dans ce travail. Cependant, nous rappellons que l'approche OLAP est très particulière et que les objets des cellules ne sont pas des données individuelles mais des données agrégées. L'interprétation d'une valeur floue dans une cellule n'est donc pas envisagée comme une distribution disjonctive de valeurs possibles, mais comme une description floue associée à un ensemble d'individus de la base source. D'autre part, l'utilisation d'une mesure de satisfiabilité permet l'obtention d'une seule valeur résultat pour mesurer à quel point la valeur répond au critère.

### 2.3.2 Prise en compte des degrés de confiance des valeurs

Pour l'équation (2.1), on envisage la possibilité de fusionner la valeur et son degré de confiance avant l'application des opérateurs de comparaison avec le critère.

Par exemple, la valeur $v$ pourrait être fusionnée avec son degré de confiance $d$ de la manière suivante : $v' = \max(1 - d, v)$. Cette méthode est classiquement utilisée, cet opérateur étant facile à calculer. Cependant, la forme des sous-ensembles flous obtenus n'est guère compréhensible après une telle opération.

Dans notre modèle, nous choisissons de comparer la valeur au critère et de considérer dans un deuxième temps le degré de confiance comme un seuil maximal. Le degré $\mu$ de chaque cellule après sélection ne peut dépasser ce seuil, ce qui empêche les cellules d'appartenir au cube avec un degré fort quand la valeur n'est pas fiable.

Pour l'équation (2.3), le problème est quelque peu différent puisque le degré $d(d_i)$ correspond au degré d'appartenance de la tranche au cube. Par exemple, si l'utilisateur a préalablement sélectionné les villes situées à l'Est, le cube est modifié le long de la dimension géographique, les degrés d'appartenance des différentes tranches au cube étant calculés selon l'appartenance (graduelle) des villes à la région Est. Ainsi, il n'est pas besoin de fusionner la valeur de la tranche avec le degré qui lui est associé.

### 2.3.3 Opérateurs d'agrégation

Mises à part quelques approches récentes [BOS 01], la plupart des systèmes de bases de données floues existants considérant le problème des requêtes floues se situent dans le cadre de données parfaitement connues [BOS 95a]. Dans notre modèle, à la fois les données et les critères revêtent des descriptions imparfaites. Pour le calcul du degré d'appartenance $\mu$ des cellules après sélection sur la mesure, les valeurs suivantes doivent être fusionnées : (i) le degré d'adéquation de la valeur au critère, (ii) le degré de confiance en la valeur, et (iii) l'ancien degré $\mu$ du cube de départ. Les propriétés ci-dessous garantissent à la fois la sémantique et les performances de calcul de notre solution.

#### Degré d'appartenance

Après une sélection, une cellule ne peut en aucun cas appartenir au cube résultat de manière plus forte qu'avant la sélection, de manière plus forte que le degré d'adéquation de la valeur au critère, ou encore de manière plus forte que le seuil fixé par le degré de confiance en cette valeur. On considère donc un opérateur qui vérifie : $\forall(x, y, z) \in [0, 1]^3$, $\mathcal{T}(x, y, z) \leq min(x, y, z)$. Ce choix conduit au fait que 0 est élément absorbant. Une t-norme est donc appropriée.

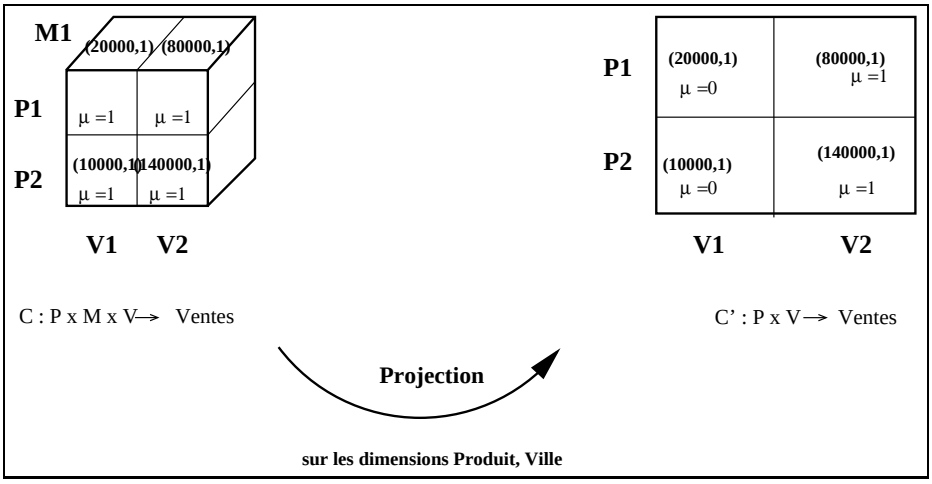


FIG. 25 – Projection

### Monotonie (décroissante)

Le résultat de l'agrégation doit décroître quand un ou plusieurs degrés décroissent : $\mathcal{T}(x', y', z') \leq \mathcal{T}(x, y, z)$ si $x' \leq x$ et $y' \leq y$ et $z' \leq z$.

### Idempotence

La propriété d'idempotence peut être souhaitable afin de ne pas modifier le degré d'appartenance d'une cellule à un cube après des sélections successives par le même critère. La seule t-norme possédant cette propriété est la t-norme min, elle doit alors être choisie pour construire $\mathcal{T}$. La solution actuelle implantée utilise cette t-norme.

Cependant, dans le cas de sélections successives par des critères différents, il peut être intéressant d'affaiblir le degré $\mu$ chaque fois qu'une sélection est effectuée. Pour cette raison, une t-norme telle le produit peut être considérée.

### Commutativité et associativité

Ces propriétés permettent au système d'optimiser les performances puisque l'ordre d'exécution des opérations n'influe pas sur le résultat et que le système de gestion de bases de données est libre d'optimiser l'ordre des calculs. Les t-normes sont commutatives et associatives, les t-normes min et produit sont donc, entre autres, appropriées dans le cadre de notre modèle.

L'opération de sélection sur les valeurs des dimensions (*slice*) utilise elle aussi la mesure de satisfiabilité $\mathcal{C}$. Un opérateur $\top$ doit être choisi. Comme précédemment, une tranche ne peut appartenir au cube résultat de manière plus forte que précédemment. Ainsi, selon que la propriété d'idempotence est requise ou non, le choix de l'opérateur se porte sur le minimum ou le produit, ces opérateurs étant communément présents dans tous les systèmes.

## 2.4 Projection

On considère ici l'opération consistant à réduire le nombre de dimensions du cube.

Par la projection du cube défini par la relation $C : D_1 \times \ldots \times D_k \rightarrow D_c$ sur les dimensions $D_{i_1}, \ldots, D_{i_n}$ ($\{D_{i_1}, \ldots, D_{i_n}\} \subseteq \{D_1, \ldots, D_k\}$) on obtient la relation $C' : D_{i_1} \times \ldots \times D_{i_n} \rightarrow D_c$

[AGR 97] introduit l'opération *destroy* pour ôter des dimensions, mais celle-ci ne peut s'appliquer que si l'on n'a qu'une seule valeur sur les dimensions candidates à l'élimination. La projection est alors définie comme la succession d'une opération de fusion et d'une opération de destruction.

Dans notre cas également, une opération est requise pour réduire le domaine pour n'avoir plus qu'une valeur. On peut envisager d'utiliser les niveaux de hiérarchie pour agréger jusqu'au niveau le plus haut (s'il existe un niveau avec une seule valeur), en utilisant une fonction d'agrégation définie selon la dimension (*e.g.* somme).

Par exemple, on considère la projection du cube $C$ sur les dimensions $D_1 \times \ldots \times D_{j-1} \times D_{j+1} \times \ldots \times D_k$ pour détruire la dimension $D_j$. $dom_C(D_j)$ est réduit à un singleton.

Le domaine actif est simplement réduit à un singleton en considérant une opération de généralisation (*roll-up*) satisfaisante (voir section suivante). C'est pourquoi on peut également considérer la projection comme le fait de se placer à un niveau de granularité élevé [MAR 98a]. La figure 25 fournit un exemple de projection.

## 2.5 Agrégation

Cette opération consiste à résumer un cube en une valeur, floue ou non. On reprend ici les opérateurs d'agrégation classiquement rencontrés, notamment dans les systèmes de gestion de bases de données relationnels, c'est-à-dire les opérateurs de comptage, minimum, maximum, somme et moyenne. Ces opérateurs sont étendus au traitement de données imparfaitement connues. Dans le contexte des bases floues, ils doivent être distingués des opérateurs utilisés précédemment pour l'agrégation des différents degrés rencontrés ($\mathcal{T}$ par exemple).

**Définition 18 (Agrégation)** *On entend par* agrégation *une opération qui, considérant un opérateur, un cube, et un critère sur les données des cellules de ce cube, renvoie un résultat flou ou numérique résumant le cube.*

Le résultat d'une agrégation est soit un nombre (pour le comptage et la moyenne), soit un sous-ensemble flou (pour le minimum, le maximum, la somme et une autre définition de moyenne).

On note alors $Ag_{cpt}(C)$ ou $Ag_{moy}(C)$ ce nombre, et $Ag_{min}(C)$ (respectivement $Ag_{max}(C)$, $Ag_{som}(C)$) les sous-ensembles flous obtenus. On note $U$ l'univers de définition du sous-ensemble flou résultat. Cet univers est le même que celui sur lequel sont définies les valeurs des éléments des cellules du cube à agréger. $Ag_{cpt}(C)(u)$ (resp. $Ag_{max}(C)(u)$, $Ag_{som}(C)(u)$) désigne le degré d'appartenance de $u \in U$ à ce sous-ensemble flou.

Nous insistons sur le fait que, pour tous les opérateurs proposés, leur application à des données parfaitement connues produira un résultat précis et certain, conformément au cas classique, c'est-à-dire qu'ils sont consistants. Les opérateurs proposés ici pour l'agrégation sont eux aussi consistants, comme l'étaient les opérateurs présentés dans les sections précédentes.

Les opérations d'agrégation qui produisent un résultat potentiellement flou utilisent l'arithmétique floue et le principe d'extension. Le résultat de l'agrégation par la fonction $\Phi$ sur les valeurs $v_i (i = 1, \ldots, n)$ résulte alors en un sous-ensemble flou de fonction d'appartenance $res$ avec :

$$\forall u \in U\ ,\ res(u) = \sup_{\{(u_1,...,u_n)|u=\Phi(u_1,...,u_n)\}} \min_{1\leq i\leq n} (v_i(u_i)) \tag{2.4}$$

Pour chaque valeur $u$ de l'univers $U$, on détermine tous les ensembles $(u_1, ..., u_n) \in U^n$ de n valeurs qui s'agrègent en $u$ par l'opérateur $\Phi$ étudié. Pour chacun de ces ensembles, on détermine le minimum des degrés d'appartenance aux valeurs des n cellules. Le degré d'appartenance de la fonction résultat en ce point $u$ de l'univers $U$ est alors le maximum de ces degrés d'appartenance.

### 2.5.1 Agrégation par comptage

Deux types de comptage sont proposés. Ils fournissent tous deux en résultat un nombre non flou, soit réel, soit entier naturel. L'un procède par $\Sigma$-comptage et somme les degrés $\mu$ des cellules, tandis que l'autre compte les cellules dont le degré $\mu$ est supérieur à un seuil $s$. Dans les deux cas, une cellule n'est prise en compte pour le comptage que si les tranches pour les valeurs associées sur toutes les dimensions appartiennent de manière suffisante au cube (degré supérieur à $deg$). Soit $C$ un cube à $k$ dimensions, on a :

$$Ag_{cpt_1} = \sum_{\overrightarrow{x}} \mu_{C'}(\overrightarrow{x}) \cdot \prod_{i=1}^{k} \delta_{deg}(d(d_i)) \tag{2.5}$$

ou

$$Ag_{cpt_2} = \sum_{\overrightarrow{x}} \delta_s(\mu_{C'}(\overrightarrow{x})) \cdot \prod_{i=1}^{k} \delta_{deg}(d(d_i)) \tag{2.6}$$

$$\text{où } \delta_{deg}(x) = \begin{cases} 1 & \text{si } x \geq deg \\ 0 & \text{sinon} \end{cases}$$

**Remarque.** Le $\Sigma$-comptage présente l'inconvénient de produire le même résultat pour des configurations très différentes. Par exemple, le même résultat est obtenu pour un cube contenant 100 cellules non vides appartenant très faiblement au cube (degré $0,01$) ou 1 seule cellules appartenant pleinement au cube. Comme proposé dans [KAC 00b], un troisième mode de comptage peut être envisagé en combinant les deux premières approches pour un $\Sigma$-comptage seuillé. Les degrés des cellules sont alors sommés pour toutes les cellules appartenant au cube avec un degré supérieur à un seuil fixé.

### 2.5.2 Agrégation par min, max et somme

Les agrégations min, max, et somme fournissent un résultat sous la forme d'un nombre flou. De même que précédemment, une cellule n'est prise en compte que si son degré $\mu$ et les degrés sur les différentes tranches attachées à cette cellule sont suffisamment grands.

On note $\mathcal{X}$ l'ensemble des cellules $\overrightarrow{x_i}$ dont les degrés $\mu$ et les degrés le long des dimensions sont considérés comme suffisants.

On applique alors l'équation (2.4) aux fonctions $\Phi = \min$, $\Phi = \max$ ou $\Phi = sum$. Chaque élément des cellules $\overrightarrow{x_i} \in \mathcal{X}$ considérées est pris en compte dans son intégralité en fusionnant pour chaque cellule l'information imprécise fournie par la valeur et son degré de confiance associé.
On calcule pour chaque cellule $\overrightarrow{x}$ la nouvelle valeur $v'(\overrightarrow{x})$ en fonction de $d(\overrightarrow{x})$ et $v(\overrightarrow{x})$. Par exemple, on a pour tout $u \in U$ $v'(\overrightarrow{x})(u) = max(1 - d(\overrightarrow{x}), v(\overrightarrow{x})(u))$.

$$\forall u \in U\ ,\ Ag_{max}(C)(u) = \sup_{\{(u_1,\ldots,u_n)|u=\max_{i=1}^{n}(u_i)\}} \min_i(v'(\overrightarrow{x_i})(u_i))$$

$$\forall u \in U\ ,\ Ag_{min}(C)(u) = \sup_{\{(u_1,\ldots,u_n)|u=\min_{i=1}^{n}(u_i)\}} \min_i(v'(\overrightarrow{x_i})(u_i))$$

$$\forall u \in U\ ,\ Ag_{som}(C)(u) = \sup_{\{(u_1,\ldots,u_n)|u=\sum_{i=1}^{n}(u_i)\}} \min_i(v'(\overrightarrow{x_i})(u_i))$$

**Exemple 14** *On considère un cube flou $C$ contenant deux cellules dont les éléments sont $(v_1, d_1)$ et $(v_2, d_2)$, définis sur un univers discret. On définit les valeurs des deux éléments avec $v_1 = \{1/130\}$, $v_2 = \{1/100, 0.9/110,\ 0.7/120\}$. Les degrés de confiance associés à ces valeurs sont $d_1 = 1$ et $d_2 = 0.8$. Ces cellules ont des degrés $\mu$ supérieurs au seuil donné. On a :*

$$\begin{aligned} v'_1 &= \{max(1-d_1,0)/100, max(1-d_1,0)/110, max(1-d_1,0)/120, \\ &\quad max(1-d_1,1)/130, max(1-d_1,0)/140,\} \\ &= \{max(0,0)/100, max(0,0)/110, max(0,0)/120, max(0,1)/130, max(0,0)/140,\} \\ &= v_1 \end{aligned}$$

$$\begin{aligned} v'_2 &= \{max(1-d_2,1)/100, max(1-d_2,0.9)/110, max(1-d_2,0.7)/120, \\ &\quad max(1-d_2,0)/130, max(1-d_2,0)/140\} \\ &= \{max(0.2,1)/100, max(0.2,0.9)/110, max(0.2,0.7)/120, max(0.2,0)/130, max(0.2,0)/140\} \\ &= \{1/100, 0.9/110,\ 0.7/120, 0.2/130, 0.2/140\} \end{aligned}$$

*Alors*

$$Ag_{max}(C) = \{1/130\}$$

*Pour le calcul, on a par exemple :*

$100 = \max(100, 100)$
*d'où*

$$\begin{aligned} Ag_{max}(C)(100) &= \max(min(v'_1(100), v'_2(100))) \\ &= min(0,1) \\ &= 0 \end{aligned}$$

$$\begin{aligned} 130 &= \max(100, 130) \\ &= \max(110, 130) \\ &= \max(120, 130) \\ &= \max(130, 130) \\ &= \max(130, 120) \\ &= \max(130, 110) \\ &= \max(130, 100) \end{aligned}$$

*d'où*

$$\begin{aligned} Ag_{max}(C)(130) &= \max(min(v'_1(100), v'_2(130)), min(v'_1(110), v'_2(130)), min(v'_1(120), v'_2(130)), \\ &\quad min(v'_1(130), v'_2(130)), min(v'_1(130), v'_2(120)), min(v'_1(130), v'_2(110)), \\ &\quad min(v'_1(130), v'_2(100))) \\ &= \max(min(0, 0.2), min(0, 0.2), min(0, 0.2), min(1, 0.2), \\ &\quad min(1, 0.7), min(1, 0.9), min(1, 1)) \\ &= 1 \end{aligned}$$

$$Ag_{min}(C) = \{1/100, 0.9/110, 0.7/120, 0.2/130\}$$

$$Ag_{som}(C) = \{1/230, 0.9/240, 0.7/250, 0.2/260, 0.2/270\}$$

### 2.5.3 Agrégation par la moyenne

Nous considérons une *mesure* de cube définie sur un domaine $U$ numérique. Une première solution est de diviser le résultat précédent de somme par la valeur de comptage :

$$Ag_{moy_1}(C)\left(\frac{u}{Ag_{cpt}(C)}\right) = Ag_{som}(C)(u) \tag{2.7}$$

Notre seconde proposition est d'étendre la proposition $f_{moy2}$ de [RUN 92] décrite page 40 :

$$Ag_{moy2}(C) = \begin{cases} \dfrac{\sum_{\vec{x}} \sum_{u \in U} v(\vec{x})(u) \cdot u}{\sum_{\vec{x}} \sum_{u \in U} v(\vec{x})(u)} & \text{si } U \text{ est dénombrable} \\ \dfrac{\sum_{\vec{x}} \int_{u \in U} v(\vec{x})(u) \cdot u \, du}{\sum_{\vec{x}} \int_{u \in U} v(\vec{x})(u) \, du} & \text{sinon} \end{cases} \tag{2.8}$$

Une autre proposition consiste à prendre en compte graduellement les valeurs des cellules en fonction de leur degré d'appartenance au cube.

$$Ag_{moy3}(C) = \begin{cases} \dfrac{\sum_{\vec{x}} \mu(\vec{x}) \sum_{u \in U} v(\vec{x})(u) \cdot u}{\sum_{\vec{x}} \mu(\vec{x}) \sum_{u \in U} v(\vec{x})(u)} & \text{si } U \text{ est dénombrable} \\ \dfrac{\sum_{\vec{x}} \mu(\vec{x}) \int_{u \in U} v(\vec{x})(u) \cdot u \, du}{\sum_{\vec{x}} \mu(\vec{x}) \int_{u \in U} v(\vec{x})(u) \, du} & \text{sinon} \end{cases} \tag{2.9}$$

**Exemple 15** *On reprend les valeurs de l'exemple précédent, elles sont au nombre de 2. On a :*

$Ag_{moy_1}(C)(\frac{230}{2}) = 1$
$Ag_{moy_1}(C)(\frac{240}{2}) = 0.9$
$Ag_{moy_1}(C)(\frac{250}{2}) = 0.7$
$Ag_{moy_1}(C)(\frac{260}{2}) = 0.2$

$Ag_{moy_1}(C)(\frac{270}{2}) = 0.2$

$$soit\ Ag_{moy_1}(C) = \{1/115, 0.9/120, 0.7/125, 0.2/130, 0.2/135\}$$

$$\begin{aligned} Ag_{moy2}(C) &= \frac{(1\cdot100+0.9\cdot110+0.7\cdot120+0.2\cdot130+0.2\cdot140)+130}{(1+0.9+0.7+0.2+0.2)+1} \\ &= 116.75 \end{aligned}$$

*On suppose* $\mu(\overrightarrow{x_1}) = 0.7$ *et* $\mu(\overrightarrow{x_2}) = 0.6$

$$\begin{aligned} Ag_{moy3}(C) &= \frac{0.7\cdot(1\cdot100+0.9\cdot110+0.7\cdot120+0.2\cdot130+0.2\cdot140)+0.6\cdot130}{0.7\cdot(1+0.9+0.7+0.2+0.2)+0.6\cdot1} \\ &= 116.26 \end{aligned}$$

*Dans le cas où les degrés $\mu$ sont très différents d'une cellule à l'autre, les valeurs des cellules à forte appartenance au cube influencent le résultat.*

*Ainsi, en considérant* $\mu(\overrightarrow{x_1}) = 0.2$ *et* $\mu(\overrightarrow{x_2}) = 1$, *on a* $Ag_{moy3}(C) = 123.375$.

**Remarques.**

1. De manière évidente, la méthode présentée par l'équation (2.9) est équivalente à celle de l'équation (2.8) quand toutes les cellules $\overrightarrow{x}$ appartiennent au cube avec des degrés $\mu(\overrightarrow{x})$ égaux. La preuve est triviale par la mise en facteur au numérateur et au dénominateur de la constante représentant le degré $\mu$ présent dans toutes les cellules.
2. La méthode doit être maniée avec précaution puisqu'elle prend en compte toutes les cellules. Dans les cas limites, elle donne le même résultat si toutes les cellules appartiennent pleinement au cube, ou alors si elles ne lui appartiennent que très faiblement et si tous les degrés $\mu(\overrightarrow{x})$ sont égaux. De même que pour le comptage, une solution intermédiaire peut être envisagée, en utilisant l'équation (2.9) et en ne considérant que les cellules dont le degré d'appartenance au cube est supérieur à un seuil fixé.

## 2.6 Généralisation

L'opération de généralisation (*Roll-Up*) consiste à généraliser un cube d'un niveau de granularité à un autre moins fin. Par exemple, on passe d'un cube décrivant les ventes par mois à un cube décrivant les ventes par trimestre. On modifie ainsi le domaine actif de la dimension le long de laquelle on généralise.

**Définition 19 (Roll-Up)** *On entend par* roll-up *une opération qui, considérant un opérateur, un cube, un critère sur les données des cellules de ce cube, et une dimension hiérarchisée, construit un cube résultat résumant le cube le long de la dimension considérée.*

Soit un cube $C$ sur lequel on applique une opération de généralisation sur la dimension $dim$, en utilisant la hiérarchie $h$ définie par la relation d'ordre floue $R$, du niveau $L_1$ au niveau $L_2$, par la fonction $\phi$ pour obtenir le cube $C'$. On a alors :

$$C \xrightarrow[\phi]{(dim,h,L_1,L_2)} C'$$

Cette opération produit le cube $C'$ où :

- le domaine actif de la dimension $dim$ a changé (les éléments ont changé avec leurs valeurs et leurs degrés),

– les degrés $\mu_{C'}$ ont changé,
– les valeurs des éléments des cellules $(v_{C'}(\overrightarrow{x}), d_{C'}(\overrightarrow{x}))$ ont changé.

Pour chaque valeur de la hiérarchie vers laquelle on généralise, on prend en compte les coefficients indiquant à quel point chacune des valeurs du niveau le plus fin se généralise en cette valeur. Dans le cas d'une hiérarchie définie par une relation floue $R$, ces coefficients sont obtenus en considérant la fermeture transitive de $R$. Dans le cas d'une hiérarchie définie par un ensemble de partitions floues, ces coefficients sont obtenus en considérant les degrés d'appartenance aux différents sous-ensembles flous de la partition considérée. On note $L_1$ l'ensemble des valeurs du niveau de granularité le plus fin et $L_2$ l'ensemble des valeurs du niveau de granularité généralisé. Les deux niveaux $L_1$ et $L_2$ sont donc inclus dans le domaine de la dimension à généraliser.

On note $\mathbf{c(a, b)}$ le coefficient de transition par généralisation entre les valeurs $a \in L_1$ et $b \in L_2$.

**Remarque.**

On note que le calcul de la fermeture transitive peut conduire à des situations contre-intuitives, par exemple si l'on considère le cas de la figure 16.b de la page 49. En effet, le calcul de la fermeture transitive résulte en un degré d'appartenance inférieur à 1 pour l'appartenance de chacune des villes au pays $USA$. Ceci est dû au fait que la hiérarchie n'est pas complète, puisqu'aucun chemin valué à 1 n'existe entre le niveau des villes et le niveau des régions. Dans ce cas, il manque un élément à la partition des régions (*Centre*) afin de rétablir des calculs intuitivement corrects.

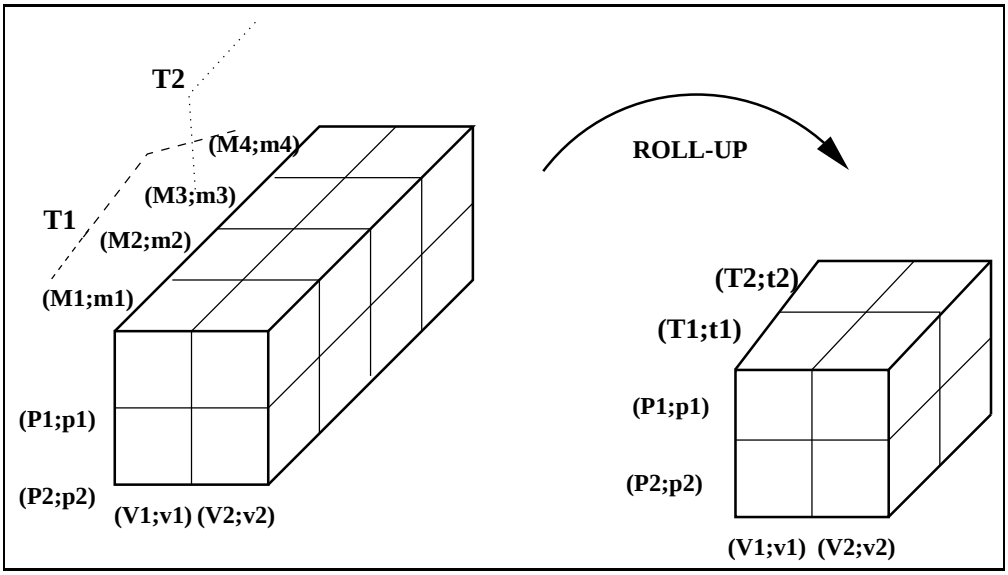


FIG. 26 – Généralisation

Les degrés associés aux cellules et aux tranches du cube $C'$ résultant d'une opération de généralisation changent en fonction des degrés précédents et des coefficients de transition.

Ainsi, pour tout $b \in L_2$, $\mu_{C'}(..., b, ...)$ s'exprime en fonction (i) des $\mu_C(..., a, ...)$ avec $a \in L_1$ et (ii) des coefficients $c(a, b)$. On définit alors

$$\mu_{C'}(..., b, ...) = \max_{a \in L_1} \min(c(a, b), \mu_C(..., a, ...))$$

Concernant les degrés des tranches, on effectue la même opération. On rappelle que chaque élément $e$ du domaine de la dimension à généraliser est un couple valeur/degré de la forme $(v(e), d(e))$. Pour tout élément $e_2$ du domaine d'arrivée de la dimension généralisée tel que $v(e_2) = b$, on calcule donc

$$d_{C'}(e_2) = \max_{\{e_1 | v(e_1)=a\}} \min(c(a,b), d(e_1))$$

**Exemple 16** *Sur la figure 26, la cellule à la position $(P1, T1, V1)$ du cube d'arrivée est obtenue en agrégeant les cellules aux positions $(P1, M1, V1)$, $(P1, M2, V1)$, $(P1, M3, V1)$ et $(P1, M4, V1)$ du cube de départ. La prise en compte de ces cellules parmi les cellules participant à la généralisation dépend des degrés $m1$, $m2$, $m3$ et $m4$ sur les tranches, des degrés d'appartenance de ces cellules au cube (par exemple, $\mu((P1, M2, V1))$ doit être supérieur au seuil fixé), et de la relation entre les mois et les trimestres ($M1, M2$, et $M3$ se généralisent en $T1$ avec un degré 1 tandis que la généralisation de $M4$ en $T1$ est plus faible). Les degrés $m1$, $m2$, $m3$ et $m4$ serviront également à calculer le degré $T1$, et les degrés $\mu$ des cellules participant à la généralisation serviront au calcul du degré $\mu((P1, T1, V1))$.*

*Comme précédemment, $v'(\overrightarrow{x})$ désigne la valeur obtenue en fusionnant la valeur $v(\overrightarrow{x})$ et son degré de confiance $d(\overrightarrow{x})$ associé.*

### 2.6.1 Généralisation par comptage

Comme nous l'avons vu précédemment, deux fonctions de comptage sont disponibles. Ces deux opérateurs sont également disponibles pour généraliser un cube par comptage, et sont modifiés afin de prendre en compte les degrés $\mu$ de toutes les cellules qui doivent participer au calcul. Ici, les seuils *deg* et $t$ jouent le même rôle que dans le cas de l'agrégation par comptage. Pour chaque $b \in L_2$, ces cellules sont celles des tranches du cube correspondant à toutes les valeurs $a \in L_1$ qui se généralisent en une valeur $b$.
On considère une t-norme $\top$, le produit par exemple, on a alors : $\forall b \in L_2$ ,

$$v_{cpt_1}(C')(..., b, ...) = \sum_{a \in L_1} \top(\mu_C(..., a, ...), c(a,b)) \cdot \prod_{i=1,...,k} \delta_t(d(d_i))$$

$$v_{cpt_2}(C')(..., b, ...) = \sum_{a \in L_1} \delta_{deg}(\top(\mu_C(..., a, ...), c(a,b))) \cdot \prod_{i=1,...,k} \delta_t(d(d_i))$$

**Remarque** Le cube $C'$ n'est pas exactement défini comme le cube précédent $C$ puisque la mesure a été modifiée et est maintenant définie sur l'univers $\mathbb{R}$ dans le premier cas et $\mathbb{N}$ dans le second cas.

### 2.6.2 Généralisation par le minimum, le maximum et la somme

Les ensembles $L_1$ et $L_2$ étant donnés, soit $a$ dans $L_1$ et $b$ dans $L_2$. On ne prend en compte que les cellules aux positions $(\ldots, a, \ldots)$ et pour lesquelles les degrés $\mu$ et les degrés sur les tranches sont satisfaisants, et qui se généralisent de manière suffisante dans la valeur $b \in L_2$. Plus précisément, pour tout $b \in L_2$, on note $\mathcal{Y}$ l'ensemble des cellules $\overrightarrow{x_i}$, $i = 1, \ldots, n$, telles que :

- la position $\overrightarrow{x_i}$ est de la forme $(..., a, ...)$ avec $a \in L_1$,
- $c(a,b)$ est supérieur au seuil $\xi$ fixé,
- $\mu(\overrightarrow{x_i})$ est supérieur au seuil correspondant *deg* fixé,
- les degrés de toutes les tranches associées à $\overrightarrow{x_i}$ sont supérieurs au seuil $t$ fixé.

Pour chaque opérateur d'agrégation $\Phi \in \{\min, \max, \sum\}$, on a alors : $\forall b \in L_2, \forall u \in U$,

$$v_{C_\Phi}(..., b, ...)(u) = \sup_{\{(u_1,...,u_n)|u=\Phi(u_1,...,u_n)\}} \min_{i=1}^{n}(v(\overrightarrow{x_i})(u_i))$$

### 2.6.3 Généralisation par la moyenne

On reprend cette fois encore l'opérateur d'agrégation défini précédemment et on l'étend pour le calcul du nouveau cube $C'$ généralisé pour la prise en compte des degrés de transition entre les valeurs des différents niveaux de granularité. On a alors pour tout $b \in L_2$ sur la dimension à généraliser :

$$v_{C_{moy2}}(..., b, ...) = \frac{\sum_{\overrightarrow{x_i} \in \mathcal{Y}} \top(\mu(\overrightarrow{x_i}), c(a,b)) \cdot \int_{u \in U} v(\overrightarrow{x_i})(u).u \; du}{\sum_{\overrightarrow{x_i} \in \mathcal{Y}} \top(\mu(\overrightarrow{x_i}), c(a,b)) \cdot \int_{u \in U} v(\overrightarrow{x_i})(u) \; du} \tag{2.10}$$

# Chapitre 3

# Manipulation des données : opérateurs binaires

L'étude des opérateurs binaires a été très peu approfondie dans les modèles de la littérature. Les opérateurs binaires sont moins définis pour répondre à des besoins spécifiques OLAP que pour étendre les opérateurs relationnels existants au contexte multidimensionnel. Ainsi, les requêtes typiques OLAP décrites dans la littérature ne font jamais appel à des opérateurs de fusion de cubes. Nous nous attachons à répertorier les opérateurs binaires existants, à les comparer pour ensuite nous interroger sur les besoins spécifiques d'opérateurs binaires des modèles classiques et étendus au flou dédiés aux analyses OLAP.

Fondé sur la théorie des ensembles, le modèle relationnel est pourvu de cinq opérateurs de base (sélection, projection, union, produit cartésien, différence) qui définissent tous les opérateurs de manipulation [ULL 88]. Les opérateurs dits *ensemblistes* sont l'union, l'intersection, la différence. Des opérateurs binaires tels la $\theta-$jointure et le produit cartésien sont également définis.

Pour présenter rapidement les problèmes posés par la représentation multidimensionnelle, prenons l'exemple de deux cubes représentant tous deux les résultats de ventes de produits dans différentes villes pour différents mois. Intuitivement, le calcul de l'union de ces deux cubes semble aisée. Il *suffirait*, se dit-on, de retenir dans le cube résultat les valeurs de cellules dans l'un ou l'autre des cubes. Cependant dans le cas où, dans les deux cubes, une valeur de cellule est définie à une même position, le problème de la définition de la valeur de la cellule du cube résultat à cette même position n'est pas aisée. Si l'on se réfère au cas relationnel, les deux valeurs doivent apparaître, comme 2 n-uplets apparaîtraient dans le cas relationnel dans la relation résultat. Dans le cas multidimensionnel, ces deux cellules doivent être combinées par une fonction définie par l'utilisateur ou le modèle, ou encore être réagrégées en utilisant la même fonction que celle qui a servi à la construction du cube (sous réserve que celle-ci ait les propriétés requises). Une autre solution consiste à considérer ces deux cellules comme incompatibles et à ne pas produire de cellule résultat à la position considérée.

Ce chapitre s'attache dans un premier temps à présenter les diverses approches de la littérature et des systèmes commerciaux. Une solution est ensuite envisagée dans le cadre du modèle proposé.

## 3.1 Opérateurs binaires multidimensionnels de la littérature

Certains modèles de la littérature ne fournissent aucun opérateur de combinaison du cube [VAS 98]. Les sections suivantes présentent les opérateurs existants.

### 3.1.1 Modèle d'Agrawal et al.

Dans [AGR 97], l'opérateur binaire de jointure (*join*) est défini. Deux cubes sont alors joints à l'aide d'une fonction $f_{merge}$ pour fusionner les valeurs de cellules. Une valeur n'apparaît que s'il existe au moins une cellule non vide dans la tranche. Selon les structures des cubes dont on fait la jointure, plusieurs cas particuliers apparaissent.

- On obtient le *produit cartésien* si aucune dimension n'est en commun.
- La combinaison de deux cubes à des niveaux de granularité différents (*associate*) est une opération spécifique OLAP asymétrique répondant aux requêtes telles *Exprimer chaque résultat de ventes mensuel comme un pourcentage des ventes trimestrielles.* Le résultat dans les cellules du cube obtenu est calculé en divisant les valeurs du cube de niveau le plus fin par les valeurs de l'autre cube (plus agrégé) en considérant la hiérarchie. On note que l'expression de cette requête en SQL est pratiquement impossible.

Les opérateurs binaires d'union, intersection, et différence sont définis à partir des autres opérateurs. Deux cubes sont union-compatibles si :

- ils ont le même nombre de dimensions
- pour chaque dimension, les domaines sont de même type dans les deux cubes

L'union est calculée en considérant une fonction d'agrégation $f_{elem}$ pour fusionner les valeurs des cellules pour chaque position applicable dans au moins l'un des deux cubes de départ afin de produire la valeur de la cellule du cube résultat. Les dimensions du cube résultat contiennent l'union des valeurs des domaines des deux cubes de départ.

L'intersection est définie sur deux cubes qui doivent être union-compatibles. On ne retient que les valeurs des dimensions qui sont présentes dans les deux cubes. Les autres positions produisent des cellules vides dans le cube résultat. Une fonction $f_{elem}$ (différente de la précédente) est utilisée pour fusionner les valeurs.

La différence est définie sur deux cubes qui doivent être union-compatibles. Cette opération est définie comme une intersection suivie d'une union avec le premier cube.

Concernant des aspects moins conceptuels, les auteurs indiquent également que l'opérateur *drill-down,* contrairement à ce que l'on pourrait penser, est un opérateur binaire. En effet, une opération de jointure (association) doit être effectuée entre le cube de niveau agrégé et le cube de niveau détaillé.

### 3.1.2 Modèle de Gyssens et al.

Dans [GYS 97], les opérations classiques, et donc les opérations binaires d'union, intersection, différence et produit cartésien, sont traduites en relationnel sur les tables relationnelles sous-jacentes au cube.

L'union, l'intersection et la différence sont définies pour des tables ayant même schéma n-dimensionnel. Le résultat a le même schéma et cette instance contient exactement le résultat donné dans le cas relationnel en considérant les deux relations de départ et l'opérateur union $\cup$, intersection $\cap$, et différence $\setminus$. Ces opérations n'interviennent donc pas directement sur une représentation multidimensionnelle.

### 3.1.3 Modèle de Datta et al.

Dans [DAT 99], les opérateurs binaires décrits sont très proches de ceux d'Agrawal. Le produit cartésien permet d'effectuer une opération entre deux cubes ayant n'importe quelle structure. Le cube résultat a pour ensemble de dimensions l'union des dimensions des deux cubes de départ. De même la mesure du cube résultat regroupe les mesures des deux cubes de départ. La jointure est un cas particulier du produit cartésien, quand une ou plusieurs dimensions sont en commun dans les deux cubes.

Deux cubes sont union-compatibles s'ils ont mêmes dimensions. L'exemple donné est l'union du cube des ventes dans les villes de l'Est avec le cube des ventes dans les villes de l'Ouest. La façon de combiner les valeurs des cellules en cas de recouvrement des valeurs des dimensions entre les deux cubes n'est pas donnée même si le cas est envisagé. Dans leur définition de l'union, les auteurs stipulent que la mesure du cube résultat est la même que les mesures des cubes de départ, il s'agirait donc de fusionner les valeurs des cellules en cas de conflit entre deux valeurs pour une même position dans le cube. Cependant, les auteurs indiquent également que les valeurs contenues dans le cube résultat correspondent à l'union des valeurs contenues dans les cubes de départ, ce qui est contradictoire. Ce modèle est donc très peu clair sur la façon dont sont combinées les valeurs des cellules dans le cas d'opérateurs binaires.

### 3.1.4 Modèle de Li et Wang

L'algèbre multidimensionnelle proposée dans [LI 96] permet d'effectuer la jointure entre une table et un cube, et l'union de deux cubes.

L'union est définie pour deux cubes ayant même schéma et au moins une dimension de domaines disjoints. Sur cette dimension, aucune valeur ne doit être partagée. Chaque cellule prend sa valeur originelle. Si une nouvelle position apparaît, alors la cellule est vide. Ainsi, chaque cellule du cube résultat a au plus une cellule d'origine. L'opération $rc$-join permet d'effectuer la jointure entre une relation $r$ dans une dimension d'un cube (construction de hiérarchies).

### 3.1.5 Modèle de Cabibbo et Torlone

Les opérateurs binaires définis dans [CAB 98] sont le produit cartésien et la jointure.
**Produit cartésien :** le cube résultat contient une entrée pour chaque paire d'entrées dans les expressions de départ avec comme mesure la juxtaposition des valeurs des deux mesures de départ.

**Jointure** (dimensions en commun) : le cube résultat a une entrée pour chaque paire d'entrées dans les expressions de départ ayant en commun les attributs de jointure. La mesure correspondante est la juxtaposition des mesures des entrées de départ.

### 3.1.6 Modèle de Munneke et al.

Dans [MUN 99], l'opérateur *join* est défini pour reconstruire les cubes fragmentés. La fragmentation est opérée pour des raisons d'optimisation du stockage physique et non pour des raisons conceptuelles. Les fragments du cube peuvent alors être stockés sur différents disques. Une erreur se produit si une position a une valeur dans les deux cubes de départ. Dans le cas où aucun cube ne contient de valeur pour une position donnée, la valeur 0 est affectée.

Ce traitement vérifiant que les deux cubes de départ sont bien disjoints est assuré par la fonction $f_{unique}$. Les fragments sont toujours *union-compatibles*, c'est-à-dire qu'ils ont même nombre de dimensions et que les domaines sont compatibles. Par exemple, une fragmentation est opérée en séparant les cubes décrivant différentes régions.

### 3.1.7 Modèle de Pedersen et al.

Les opérateurs fournis dans le modèle introduit par Pedersen et al. sont très proches des opérateurs relationnels [PED 99b] : union, différence et jointure.

Les objets multidimensionnels étant exprimés à partir de relations, les résultats de ces opérateurs binaires sont strictement équivalents aux résultats qui seraient obtenus avec l'algèbre relationnelle. Pour chaque opérateur multidimensionnel proposé, le système applique l'opérateur relationnel correspondant, en ôtant ensuite les données dupliquées.

Cependant, dans le cas de l'union par exemple, le problème se pose quand deux objets multidimensionnels sont combinés par une union et qu'à une position donnée les deux objets de départ contiennent une valeur de mesure (table des faits) différente. Le passage au relationnel donne une instance de relation contenant deux valeurs possibles pour les mêmes valeurs de dimensions, violant donc la dépendance liant l'ensemble des dimensions du cube à la mesure.

### 3.1.8 Systèmes commerciaux

Dans cette section, les principaux systèmes commerciaux sont passés en revue. [PEN 01] sépare les systèmes *hypercubes* des systèmes *multicubes*. Les premiers sont plus compréhensibles pour l'utilisateur, les seconds sont plus efficaces. Dans les deux cas, des combinaisons de mesures sont envisagées.

#### Oracle Express

Dans Express, des requêtes inter-cubes sont possibles, notamment pour combiner les mesures du cube [SER c]. Ces fonctionnalités ne sont pas vraiment faites pour un travail de sélection sur les dimensions ou les cellules.

Par exemple, typiquement, les cubes *units, sales* et *price* sont combinés pour obtenir le cube **NouveauCube = sales - units * price** décrivant les bénéfices. Les cubes combinés n'ont pas forcément exactement les mêmes dimensions. Express repère seul la ou les dimensions en commun pour construire le cube résultat.

Dans cet exemple, les cubes sont définis de la manière suivante :
- NouveauCube : <Mois,Produit,Ville>
- Sales : <Mois,Produit,Ville>
- Unités : <Mois,Produit,Ville>
- Prix : <Mois,Produit>

Dans ce système, il n'existe pas d'opérateur de type union, intersection, ou différence mais seulement des opérateurs arithmétiques.

On note que les fonctionnalités OLAP sont maintenant intégrées dans Oracle 9i, qui permet le partitionnement physique des données multidimensionnelles.

### Crystal Decisions - Holos

[HOL ] propose une solution OLAP composite (*Compound OLAP*) qui autorise la combinaison de bases OLAP multidimensionnelles (MOLAP) ou relationnelles (ROLAP) pour lier des données physiques pour la création de vues. La possibilité est donnée de construire une vue à partir d'autres vues sans limitation de profondeur.

### MS SQL Server - Analysis Services

SQL Server fournit la possibilité de partitionner les cubes (*Partition Wizard*). L'administrateur peut s'apercevoir par exemple que 80% des requêtes concernent les ventes récentes de produits et choisir alors de séparer le stockage physique des ventes des périodes récentes dans un cube et celles des périodes plus anciennes dans un autre. Les optimisations au niveau des performances sont alors dues à l'utilisation de plusieurs disques durs de stockage, et plusieurs processeurs (chaque partition est traitée par un processus [SER b]). Cependant, il s'agit toujours du même cube logique, traité physiquement de manière transparente pour l'analyste, même si le stockage physique est séparé.

La possibilité de combiner des cubes est fournie par l'utilisation de *cubes virtuels*. Ces cubes ne sont pas stockés physiquement mais sont l'équivalent des vues dans le modèle relationnel. Ils sont calculés à la volée.

Le langage *Multidimensional Expressions* (MDX) proposé permet l'interrogation des cubes et fournit les primitives de requêtes multidimensionnelles [NOL 99]. La définition des cubes dans la partie OLAP de SQL Server définit un cube avec un ensemble de dimensions et un ensemble de mesures. Ainsi, l'optique multi-cube est appréhendée, au moins pour des cubes ayant même structure, par le biais de mesures complexes qui sont des dimensions à plusieurs membres. Par exemple, le cube jouet *Sales* contient les mesures suivantes : *unit sales, store cost, store sales, sales count,* et *store sales.* Une nouvelle mesure peut être calculée à partir des précédentes en utilisant les opérateurs arithmétiques. Par exemple, on calcule l'expression représentant le pourcentage de profit, qui s'exprime de la manière suivante :

$$(Measures.[StoreSales] - Measures.[StoreCost])/Measures.[StoreCost]$$

Ce même type d'expression peut être utilisé pour calculer des valeurs combinant différents niveaux de hiérarchie sur l'une des dimensions.

### IBM DB2 OLAP Server

Dans ce système il est conseillé de partitionner la base multidimensionnelle pour améliorer les performances [SER a].

## 3.2 Discussion

Nous proposons de diviser ces approches selon qu'elles résultent plutôt de traitements algébriques, ou de traitements arithmétiques sur les données.

Les modèles de la littérature, souvent issus directement de l'approche relationnelle, abordent plutôt les opérateurs binaires avec des opérateurs algébriques et travaillent sur les positions de cellules.

De leur côté, les systèmes commerciaux combinent les cubes à l'aide d'opérateurs arithmétiques, par exemple pour calculer les bénéfices, et travaillent donc plus au niveau de la *valeur de cellule*. Dans ce cas, considérer la somme de deux cubes revient à faire l'union des cellules si les cellules vides ne sont pas prises en compte et que les cubes de départ sont disjoints.

D'après [DAT 99], unir deux cubes qui décriraient les ventes à l'Est et à l'Ouest ne reflète pas un stockage efficace, mais peut arriver si les deux cubes proviennent de différentes opérations préalables et sont matérialisées pour un temps très court afin d'être combinés. L'auteur semble donc faire allusion à des traitements complexes nécessitant plusieurs opérations successives sur des cubes constituant des résultats intermédiaires. Pourtant, cette approche de partitionnement est celle préconisée par de nombreux systèmes pour améliorer les performances.

Cependant, nous écartons l'étude des opérations de fragmentation et de gestion des partitions qui sont orientées vers la gestion physique des gros cubes sans influer sur les requêtes des analystes. En effet, ces méthodes permettent, par exemple, de prendre en compte la fréquence des requêtes si 80% d'entre elles n'accèdent qu'à une partie du cube, et de partitionner ce cube en deux cubes disjoints. Cependant, ces méthodes ne sont en rien liées à la sémantique des manipulations typiques OLAP dont l'utilisateur a besoin, mais simplement liés à des problématiques de performances, transparentes pour l'utilisateur.

Les cubes virtuels, en revanche, sont intéressants car calculés pour simplifier ou combiner des cubes entre eux. Ils sont apparentés aux vues des modèles relationnels.

Cependant, les langages d'interrogation de cubes sont bien souvent dédiés à l'utilisation d'un seul cube.

Pour tous les opérateurs multidimensionnels, une attention particulière doit être apportée au traitement sémantique de la mesure. En effet, la compatibilité calculatoire (même type) ne suffit pas car la plupart des cubes ont une mesure numérique. Par exemple, la pertinence des fonctions de combinaison des valeurs des cellules dépend en grande partie de la fonction d'agrégation utilisée pour la construction du cube. Ainsi, sommer des valeurs obtenues en résumant des données sources par des moyennes n'a aucun sens. En ce qui concerne l'union, nous proposons une alternative aux fonctions de type $f_{unique}$ qui empêchent le calcul de l'union de deux cubes si une position a une valeur source dans chacun des cubes de départ. En utilisant la théorie des possibilités, on peut en effet se permettre de garder l'ensemble des valeurs possibles trouvées dans les cubes de départ.

En ce qui concerne la jointure, les systèmes existants choisissent soit de juxtaposer les valeurs de mesures, soit de fusionner ces valeurs.

Pour résumer, l'apparition d'opérateurs binaires dans les modèles et systèmes existants est liée soit à la volonté d'étendre les opérateurs du modèle relationnel, soit à la volonté d'améliorer les performances.

D'ailleurs, dans [AGR 97] les auteurs reconnaissent plutôt que les opérateurs ont été introduits pour coller au relationnel. Dans les requêtes proposées en exemple, seule l'opération binaire *associate* est utilisée pour traiter des données à différents niveaux de hiérarchie (pas d'union ou autre opérateur binaire).

## 3.3 Proposition

Nous nous proposons de recueillir et définir les opérateurs binaires nécessaires aux analyses OLAP. Nous nous plaçons à un niveau conceptuel et nous ne nous attachons pas

aux aspects liés au stockage physique et à la fragmentation des cubes.

Deux optiques sont prises en compte, d'une part pour fournir aux analystes les outils nécessaires aux manipulations OLAP, et d'autre part pour permettre aux outils de fouille de données de travailler dans un environnement multi-cube.

La première question à se poser est donc la nature des cubes que l'on pourrait rencontrer dans les bases multidimensionnelles (magasins de données). Or les méthodes de construction de magasins de données sont très peu étudiées dans la littérature.

Dans notre modèle, on définit aussi la notion de *union-compatible* de la manière suivante : deux cubes sont union-compatibles s'ils ont mêmes dimensions et même(s) mesure(s).

Finalement, les opérateurs algébriques ne semblent pas avoir leur place. La construction du cube doit prendre en compte de telles manipulations (on laisse de côté les aspects physiques liés aux performances). Il ne reste alors qu'à définir les moyens de combiner de manière arithmétique les valeurs de cellules, tout en prenant en compte des aspects liés aux domaines des dimensions.

En fait, le modèle relationnel, basé sur la théorie des ensembles, nécessite des opérateurs ensemblistes. Les faits, les dimensions, et les hiérarchies sont séparés dans différentes tables, liées par des clés primaires et étrangères. C'est pour cette raison que les opérateurs binaires (jointure notamment), sont absolument nécessaires et très utilisés dans ce modèle. Cependant, le modèle multidimensionnel ne repose pas sur les mêmes principes, comme le souligne [HYP 00]. Dans un modèle de stockage OLAP multidimensionnel, les données sont des valeurs indexées par leur position sur les dimensions. Le domaine est connu et pratiquement statique, le nombre de points dans l'espace multidimensionnel est connu, contrairement au modèle relationnel où les données ne sont pas stockées selon les dimensions.

Les données à analyser étant traitées par valeurs (cellules) et non comme un ensemble de n-uplets, il s'agit plutôt de combiner des valeurs numériques que des ensembles de n-uplets, nous nous focalisons donc sur les opérateurs arithmétiques.

La construction de magasins de données efficaces pour l'analyse en ligne à la fois par navigation dans les données et par des moyens de découverte automatique est donc le principal enjeu, plutôt que la définition d'opérateurs binaires.

Toutefois, on peut imaginer de nombreuses requêtes qui nécessitent l'emploi de tels opérateurs non arithmétiques et qui pourtant ne sont jamais citées dans les requêtes typiques OLAP, notamment liées aux opérations de différence. Par exemple, on souhaite sélectionner les produits qui se sont vendus à l'Est mais pas à l'Ouest.

Dans le cadre du modèle proposé ici, nous définissons deux types d'opérateurs binaires : l'union et l'intersection. Ces opérateurs fournissent les outils nécessaires à l'application de requêtes complexes conjonctives et disjonctives sur un cube flou.

## 3.4 Union et intersection de cubes flous

On considère ici les opérations d'union et d'intersection de deux cubes. Ces opérations ne peuvent s'appliquer que sur des cubes construits selon les mêmes dimensions, même si les domaines actifs sont différents.

**Définition 20 (Structure de cube)** *Soit 2 cubes $C_\alpha$ et $C_\beta$ définis sur $D_1^\alpha \times \ldots \times D_k^\alpha$ et $D_1^\beta \times \ldots \times D_l^\beta$ de domaines $dom(D_i^\alpha)$ $i = 1, \ldots, k$ et $dom(D_i^\beta)$ $i = 1, \ldots, l$.*

*On dit que ces 2 cubes ont même structure si :*

1. $k = l$,
2. $\forall i = 1, \ldots, k$, *il existe* $j \in \{1, \ldots, k\}$ *tel que* $D_i^\alpha = D_j^\beta$. *Les domaines actifs ne sont pas nécessairement les mêmes,*
3. *les univers de définition de la mesure sont compatibles.*

Les opérations d'union et d'intersection de cubes nécessitent les opérateurs d'union et intersection de sous-ensembles flous, ainsi qu'une t-norme $\top$ et une t-conorme $\perp$ (voir annexe A).
On note $f_A$ (resp. $f_B$ et $f_C$) la fonction d'appartenance de $A$ (resp. $B$ et $C$). On construit l'union $C$ de deux sous-ensembles $A$ et $B$ ($C = A \cup_\perp B$) sur l'univers $X$ par

$$\forall x \in X \quad f_C(x) = \perp(f_A(x), f_B(x))$$

On construit l'intersection $C$ de deux sous-ensembles $A$ et $B$ ($C = A \cap_\top B$) sur l'univers $X$ par

$$\forall x \in X \quad f_C(x) = \top(f_A(x), f_B(x))$$

**Définition 21 (Union)** *Soit 2 cubes* $C_\alpha$ *et* $C_\beta$ *de même structure. On note* $C_\gamma = C_\alpha \bigcup C_\beta$ *le cube union défini sur* $D_1^\alpha \times \ldots \times D_k^\alpha$ *avec* $\forall i = 1, \ldots, k$, $aVdom_{C_\gamma}(D_i) = aVdom_{C_\alpha}(D_i) \cup aVdom_{C_\beta}(D_i)$.

*Les éléments du cube* $C_\gamma$ *sont calculés à partir des éléments des cubes* $C_\alpha$ *et* $C_\beta$.
$v(C_\gamma)(\overrightarrow{x}) = \cup_\perp(v_\alpha(\overrightarrow{x}), v_\beta(\overrightarrow{x}))$
$d(C_\gamma)(\overrightarrow{x}) = \perp(d_\alpha(\overrightarrow{x}), d_\beta(\overrightarrow{x}))$
*et* $\mu_\gamma(\overrightarrow{x}) = \perp(\mu_\alpha(\overrightarrow{x}), \mu_\beta(\overrightarrow{x}))$
*où* $\perp$ *est une t-conorme.*

*Les degrés associés aux différentes tranches du cube, c'est-à-dire les nouveaux degrés sur les dimensions sont calculés en fonction des degrés des deux cubes dont on calcule l'union :*
$\forall i \in [1, k]$, $\forall d_i \in aVdom(D_i)$ : $d^\gamma(d_i) = \perp(d^\alpha(d_i), d^\beta(d_i))$

**Remarque** Afin de rendre les calculs possibles, on admet que si la position $\overrightarrow{x}$ n'est pas valide c'est-à-dire si $E(C)(\overrightarrow{x}) = \emptyset$[10], alors on a $v_C(\overrightarrow{x})(u) = 0$ pour tout $u$ de l'univers $U$ considéré et $d_C(\overrightarrow{x}) = 0$. Ainsi, une valeur seulement présente dans l'un des deux cubes sera reprise dans le cube résultat pour l'union et ne sera pas présente dans le domaine actif de la dimension dans le cas de l'intersection. Ceci résulte du fait que 0 est élément neutre des t-conormes et élément absorbant des t-normes.

Les degrés associés aux différentes tranches du cube, c'est-à-dire les nouveaux degrés sur les dimensions sont calculés en fonction des degrés des deux cubes dont on calcule l'union :
$\forall i \in [1, k]$, $\forall d_i \in aVdom(D_i)$ : $d^\gamma(d_i) = \perp(d^\alpha(d_i), d^\beta(d_i))$

On a donc $\forall(\overrightarrow{x}) \in (D_1 \times \ldots \times D_k)$,
$E(C_\gamma)(\overrightarrow{x}) =$

[10] La notation usuelle des cellules nulles (vides) est plutôt la suivante : $\perp$. Cependant, nous ne l'utilisons pas pour ne pas induire de confusion avec l'opérateur $\perp$ définissant une t-conorme.

$$\begin{cases} \emptyset & \text{si la position } (\overrightarrow{x}) \text{ n'est valide dans aucun cube} \\ E(C_\alpha)(\overrightarrow{x}) & \text{si la position } (\overrightarrow{x}) \text{ n'est valide que dans } C_\alpha \\ \quad (\text{resp. } E(C_\beta)(\overrightarrow{x})) & \quad (\text{resp. } C_\beta) \\ (\cup_\perp(v_\alpha(\overrightarrow{x}), v_\beta(\overrightarrow{x})), \perp(d_\alpha(\overrightarrow{x}), d_\beta(\overrightarrow{x}))) & \text{si la position } (\overrightarrow{x}) \text{ est valide dans les 2 cubes} \end{cases}$$

avec $\mu_\gamma(\overrightarrow{x}) =$

$$\begin{cases} \perp(0,0) = 0 & \text{si la position } (\overrightarrow{x}) \text{ n'est valide dans aucun cube} \\ \perp(\mu_\alpha(\overrightarrow{x}), 0) = \mu_\alpha(\overrightarrow{x}) & \text{si la position } (\overrightarrow{x}) \text{ n'est valide que dans } C_\alpha \\ \quad (\text{resp. } \mu_\beta(\overrightarrow{x})) & \quad (\text{resp. } C_\beta) \\ \perp(\mu_\alpha(\overrightarrow{x}), \mu_\beta(\overrightarrow{x})) & \text{si la position } (\overrightarrow{x}) \text{ est valide dans les 2 cubes} \end{cases}$$

**Définition 22 (Intersection)** *Soit 2 cubes $C_\alpha$ et $C_\beta$ de même structure. On note $C_\gamma = C_\alpha \cap C_\beta$ le cube intersection défini sur $D_1^\alpha \times \ldots \times D_k^\alpha$ avec $\forall i = 1, \ldots, k$, $aVdom_{C_\gamma}(D_i) = aVdom_{C_\alpha}(D_i) \cap aVdom_{C_\beta}(D_i)$.*

*Les éléments du cube $C_\gamma$ sont calculés à partir des éléments des cubes $C_\alpha$ et $C_\beta$.*
$v(C_\gamma)(\overrightarrow{x}) = \cap_\top(v_\alpha(\overrightarrow{x}), v_\beta(\overrightarrow{x}))$
$d(C_\gamma)(\overrightarrow{x}) = \top(d_\alpha(\overrightarrow{x}), d_\beta(\overrightarrow{x}))$
*et* $\mu_\gamma(\overrightarrow{x}) = \top(\mu_\alpha(\overrightarrow{x}), \mu_\beta(\overrightarrow{x}))$
*où $\top$ est une t-norme.*

*Les degrés associés aux différentes tranches du cube, c'est-à-dire les nouveaux degrés sur les dimensions sont calculés en fonction des degrés des deux cubes dont on calcule l'intersection :*
$\forall i \in [1,k]$, $\forall d_i \in aVdom(D_i)$ : $d^\gamma(d_i) = \top(d^\alpha(d_i), d^\beta(d_i))$

*On a donc :*

$\forall(\overrightarrow{x}) \in (D_1 \times \ldots \times D_k)$,
$E(C_\gamma)(\overrightarrow{x}) =$

$$\begin{cases} \emptyset & \textit{si la position } (\overrightarrow{x}) \textit{ n'est pas valide dans les 2 cubes} \\ (\cap_\top(v_\alpha(\overrightarrow{x}), v_\beta(\overrightarrow{x})), \top(d_\alpha(\overrightarrow{x}), d_\beta(\overrightarrow{x}))) & \textit{si la position } (\overrightarrow{x}) \textit{ est valide dans les 2 cubes} \end{cases}$$

*avec* $\mu_\gamma(\overrightarrow{x}) =$

$$\begin{cases} 0 & \textit{si la position } (\overrightarrow{x}) \textit{ n'est pas valide dans les 2 cubes} \\ \top(\mu_\alpha(\overrightarrow{x}), \mu_\beta(\overrightarrow{x})) & \textit{si la position } (\overrightarrow{x}) \textit{ est valide dans les 2 cubes} \end{cases}$$

## 3.5 Choix des opérateurs

Deux remarques principales sont apportées aux choix faits dans cette partie de notre travail.

D'une part, les deux opérateurs de combinaison de cubes flous sont introduits principalement dans le but de répondre à des requêtes complexes. Dans ce cas, les cubes à fusionner, issus d'un même cube, ont les mêmes éléments. La fusion concerne donc principalement les degrés, les valeurs des cellules devant être maintenues. Les t-normes et t-conormes choisies pour fusionner les valeurs et les degrés sont donc choisies plutôt idempotentes (min et max).

D'autre part, le choix des noms des opérateurs (*union* et *intersection*), est lié au but dans lequel ils sont définis (requêtes complexes). Le cube résultat, hors mesure, contient

bien l'union ou l'intersection des valeurs des domaines actifs des dimensions des cubes de départ. Il représente de manière graduelle les entités qui sont soit *à la fois* dans les deux cubes, soit *au moins* dans l'un des deux. On rappelle que la mesure n'est pas issue directement des données sources mais *construite* par agrégation. Il n'est donc pas contre-intuitif qu'elle soit prise en compte différemment au moment de l'union des valeurs sur les autres dimensions, elles directement issues de la relation de départ dans l'entrepôt de données avant construction du magasin multidimensionnel.

# Chapitre 4

# Requêtes multidimensionnelles floues

Les systèmes multidimensionnels sont dédiés à la manipulation de données multidimensionnelles à l'aide de requêtes complexes. En outre, les systèmes de fouille de données qui seront greffés sur le modèle proposé nécessitent de très nombreuses requêtes sur la base. Pour ces raisons, nous étudions en détail dans ce chapitre les propriétés vérifiées par les opérateurs introduits. Ces opérateurs sont tous clos[11], c'est-à-dire que toute requête sur un hypercube a pour résultat un hypercube. La combinaison d'opérateurs est donc possible. Le but de cette étude est la définition d'une algèbre multidimensionnelle floue.

Le plan du chapitre est le suivant : dans une première partie, un langage de manipulation est détaillé et illustré par des exemples. Les propriétés vérifiées par les opérateurs unaires et leurs combinaisons sont ensuite présentées. Enfin, les moyens de construire des requêtes complexes et de les optimiser sont décrits.

## 4.1 Langage de manipulation

On définit un langage de manipulation des données, dont la syntaxe est donnée ci-dessous.

- **Définition et modification des critères et quantificateurs -**
  `CREATE <NOM><VALEUR>`, `SET <NOM><VALEUR>` où `<VALEUR>` définit un sous-ensemble flou.
- **Définition et modification des domaines des dimensions -**
  `CREATE <DOMAINE><VALEUR><DEGRE>`,
  `INSERT <DOMAINE><VALEUR><DEGRE>`,
  `DELETE <DOMAINE><VALEUR><DEGRE>` où `<VALEUR>` est un sous-ensemble flou.
- **Définition des dimensions -**
  `CREATE <DIMENSION><DOMAINE><HIERARCHIES_R><HIERARCHIES_P>`
  où `<HIERARCHIES_R>` et `<HIERARCHIES_P>` sont des listes de relations floues et de relation de précédence sur les partitions floues définies par les valeurs du domaine.
- **Définition d'un cube -**
  `CREATE <CUBE><DIMENSIONS>` où `DIMENSIONS` est un ensemble de dimensions.

[11] à l'exception de l'agrégation. Cependant cette opération peut être assimilée à une opération résultant en un cube à une seule cellule sur le niveau *ALL* pour chaque dimension.

- **Sélection sur les cellules -**
  `SelCells <CUBE><Pred>` où `<Pred>` est un prédicat défini par une fonction d'appartenance qui correspond au critère de sélection. Ce prédicat peut être un prédicat classique et dans ce cas, les valeurs de l'univers considéré répondent ou non au critère, avec des degrés nuls ou égaux à 1. Le critère peut également être un critère flou, l'adéquation des valeurs de l'univers considéré au prédicat est alors graduelle, avec des degrés d'adéquation variant de 0 à 1.
- **Sélection sur les dimensions -**
  `SelDims <CUBE><DIMENSION><Pred>`
- **Projection -**
  `Proj <CUBE><DIMENSION_TO_DESTROY>`
- **Comptage par sommation des degrés -**
  `COUNT_SUM <CUBE><SEUIL_DIM>`
- **Comptage des cellules de degré supérieur au seuil -**
  `COUNT_THRESHOLD <CUBE><SEUIL_DIM><SEUIL_CELL>`
- **Généralisation -**
  `RUp <CUBE><DIMENSION><HIERARCHY>`
- **Résumé par quantificateur -**
  `RESUME <CUBE><PREDICAT><CPT>` où `CPT` détermine le type de comptage (`COUNT_SUM` ou `COUNT_THRESHOLD`)
- **Résumé par le meilleur quantificateur -**
  `RESUME_BEST <CUBE><PREDICAT><CPT>`
- **Création de vues -**
  `CREATE VIEW <CUBE> AS <REQUETE>` où `<REQUETE>` est définie par une des opérations précédentes. Cette vue peut ensuite être réutilisée comme tout cube.

**Exemple** On considère par exemple le cube $VENTES$ défini sur les trois dimensions $PRODUIT$, $MOIS$, $VILLE$. Les requêtes suivantes vont sélectionner les ventes *faibles* et les villes situées à l'Est, puis compter les cellules du cube dont le degré $\mu$ est supérieur au seuil fixé à $0,5$. Le premier degré est le degré minimal requis sur les tranches pour la prise en compte d'une cellule dans le comptage.

```
SelCells VENTES faible
SelDims VENTES VILLE Est
COUNT_THRESHOLD VENTES 0 0,5
```

On notera que ces requêtes peuvent être combinées, par exemple par l'expression suivante :

```
COUNT_THRESHOLD (SelDims (SelCells VENTES faible) VILLE Est) 0 0,5
```

Le même résultat peut être obtenu en utilisant des vues, comme suit :

```
CREATE VIEW C1 (SelCells VENTES faible)
CREATE VIEW C2 (SelDims C1 VILLE Est)
COUNT_THRESHOLD C2 0 0,5
```

## 4.2 Combinaisons de requêtes unaires multidimensionnelles

Nous étudions ici les méthodes de réécriture de requêtes et les règles applicables à notre contexte. Les propriétés sont presque les mêmes que dans le cadre relationnel et

permettent les optimisations [ULL 88]. Afin de simplifier les notations, le langage présenté précédemment est utilisé.

### Cubes flous équivalents

Un cube flou est défini comme une application donc, de même que dans le cadre des bases de données relationnelles, deux définitions de l'équivalence sont possibles, la première s'appuyant sur la structure des cubes, et la seconde sur le contenu. Dans la suite, nous considérons que deux cubes flous sont *équivalents* si les conditions suivantes sont vérifiées :

- les 2 cubes sont définis sur les mêmes dimensions,
- les domaines actifs de toutes les dimensions sont les mêmes (mêmes éléments -valeurs et degrés) ;
- les 2 cubes contiennent exactement les mêmes cellules (mêmes éléments, mêmes degrés $\mu$).

On note que ces cubes sont presque égaux, seul l'ordre des éléments des dimensions pouvant différer.

## 4.2.1 Généralisations successives

Cette section est dédiée à l'application de plusieurs opérations de généralisation successives (*roll-Up*). Cet opérateur sert à l'agrégation des données depuis un niveau de granularité à un autre plus élevé. Par exemple, un cube décrivant les résultats de ventes de produit au niveau des villes peut être généralisé au niveau des régions.

Nous étudions ici des opérations de généralisation successives sur une même dimension, pour la même hiérarchie définie par relation floue. Soit :

- *dim* la dimension à agréger,
- $R$ la relation d'ordre floue définissant la hiérarchie sur cette dimension (on note $f_R$ sa fonction et $f_{R_T}$ la fonction de sa fermeture transitive),
- $L_1$, $L_2$ et $L_3$ les différents niveaux d'agrégation, et
- $\phi$ la fonction d'agrégation utilisée pour fusionner les valeurs des cellules.

Le but est de comparer les opérations (4.1) et (4.2) et d'étudier les conditions suffisantes pour obtenir des cubes équivalents.

$$C \xrightarrow[\phi]{(dim,R,L_1,L_2)} C' \xrightarrow[\phi]{(dim,R,L_2,L_3)} C'' \tag{4.1}$$

$$C \xrightarrow[\phi]{(dim,R,L_1,L_3)} C''' \tag{4.2}$$

### Calcul des degrés

Nous détaillons ci-dessous les moyens de calculer un cube depuis n'importe quel niveau $L_i$ jusqu'à n'importe quel autre niveau $L_j$ tel que $i < j$. En considérant $\{a_k\}$ les valeurs de la dimension à agréger pour le niveau $L_i$ et $\{b_l\}$ les valeurs pour le niveau $L_j$, on a :

$$\forall b_l \in L_j,\ d(b_l) = \max_{a_k \in L_i} \min(d(a_k), f_{R_T}(a_k, b_l))$$

$$\forall b_l \in L_j,\ \mu(\ldots, b_l, \ldots) = \max_{a_k \in L_i} \min(\mu(\ldots, a_k, \ldots), f_{R_T}(a_k, b_l))$$

On considère l'équation (4.1), on a :

$$\forall c_n \in L_3,\ d(c_n) = \max_{b_l \in L_2} \min(\max_{a_k \in L_1} \min(d(a_k), f_{R_T}(a_k, b_l)), f_{R_T}(b_l, c_n))$$

Pour l'équation (4.2), on a :

$$\forall c_n \in L_3,\ d'(c_n) = \max_{a_k \in L_1} \min(d(a_k), f_{R_T}(a_k, c_n))$$

Or,

$$f_{R_T}(a_k, c_n) = \max_{b_l \in L_2} \min(f_{R_T}(a_k, b_l), f_{R_T}(b_l, c_n))$$

D'où

$$d'(c_n) = \max_{a_k \in L_1} \min(d(a_k), \max_{b_l \in L_2} \min(f_{R_T}(a_k, b_l), f_{R_T}(b_l, c_n)))$$

On a donc $d'(c_n) = d(c_n)$ (voir démonstration en annexe D).

Les mêmes calculs valent pour les degrés $\mu$.

**Valeurs des cellules**

La fonction $\phi$ utilisée pour fusionner les valeurs $v(\overrightarrow{x})$ des cellules $\overrightarrow{x}$ doit être **associative** afin d'obtenir des cubes équivalents avec les deux équations (4.1) et (4.2).

Par exemple, une fonction de comptage *cpt* ne peut être utilisée puisque dans ce cas, le cube $C$ contient des valeurs floues alors que le cube $C'$ obtenu par généralisation contient des nombres non flous correspondant au nombre de cellules dans le cube $C$. La propriété d'associativité garantit que les valeurs obtenues directement en calculant le cube du niveau $L_1$ au niveau $L_3$ sont les mêmes qu'en calculant d'abord le cube au niveau intermédiaire $L_2$.

Les propriétés des fonctions d'agrégation sont décrites dans [GRA 97] pour la construction de cubes et le calcul de super-agrégats (agrégats d'agrégats). Les fonctions d'agrégation peuvent être *distributives* (*e.g. comptage*, *min*, *max*), *algébriques* (*e.g. moyenne*), ou *holistiques* (*e.g. mediane*).

On considère un ensemble de valeurs décrites à l'aide de deux attributs $\{X_{ij} | i = 1, ..., I \text{ et } j = 1, ..., J\}$.

- **fonctions distributives.**
  Une fonction d'agrégation $F$ est dite *distributive* s'il existe une fonction $G$ telle que :

$$F(\{X_{ij} | i = 1, ..., I \text{ et } j = 1, ..., J\}) = G(\{F(\{X_{ij} | i = 1, ..., I\}) | j = 1, ..., J\})$$

  Dans le cas de fonctions associatives, on a $F = G$, par exemple pour $F = \min$, $F = \max$, $F = sum$. Pour $F = comptage$, on a $G = somme$.

- **fonctions algébriques.**
  Une fonction d'agrégation $F$ est dite *algébrique* s'il existe un ensemble de valeurs obtenues par une fonction $G$ et une fonction $H$ tels que :

$$F(\{X_{ij}|i=1,...,I \text{ et } j=1,...,J\}) = H(\{G(\{X_{ij}|i=1,...,I\})|j=1,...,J\})$$

  Par exemple, les fonctions $F$ de calcul de la moyenne, des $N$ plus petites valeurs ($minN$), ou encore des $N$ plus grandes ($maxN$) sont des fonctions algébriques. Pour la moyenne, la fonction $G$ stocke les sommes et les cardinalités et la fonction $H$ additionne ces composants et calcule la division afin d'obtenir le résultat. Le résultat intermédiaire est donc stocké dans un espace de taille fixe.
- **fonctions holistiques.**
  Une fonction est dite *holistique* s'il n'est pas possible, en stockant un ensemble de résultats intermédiaires de taille fixe, de calculer des agrégats d'agrégats. Par exemple, les fonctions de calcul de la médiane ou du rang sont holistiques puisqu'elles nécessitent le parcours de toute la base détaillée.

Les deux premières catégories de fonctions permettent des optimisations tandis que la troisième requiert le calcul direct du cube généralisé sans aucun moyen d'utiliser les niveaux intermédiaires. Par exemple, la valeur *minimale* peut être facilement calculée à partir des résultats du cube intermédiaire, puisque la fonction min est associative. Pour le comptage, la *somme* des valeurs obtenue au niveau intermédiaire doit être calculée, tandis que le calcul de la *moyenne*, s'il veut pouvoir s'effectuer de manière optimisée, nécessite la gestion de deux valeurs (*comptage* et *somme*).

Pour conclure, les résultats suivants sont avancés, le cas flou étant équivalent au cas classique.

- les deux équations (4.1) et (4.2) ne produisent des cubes équivalents que si la fonction $\phi$ utilisée est associative.
- En utilisant une fonction $\phi'$ pour le calcul du cube $C''$ à partir de $C'$, l'ensemble des fonctions distributives peut être optimisé, $\phi' = \phi$ étant le cas particulier des fonctions associatives.
- Les fonctions algébriques peuvent être optimisées sous réserve de la gestion de plusieurs valeurs dans les cellules du cube, d'où la nécessité de mesures *complexes* avec un schéma de cube de la forme $D_1 \times \ldots \times D_k \rightarrow D_l \times ... \times D_m \times [0,1]$. Les fonctions $\phi$ doivent permettre la prise en compte des mesures $D_l, ..., D_m$.
- Aucune possibilité de calcul indirect n'existe pour les fonctions dites *holistiques*, qui nécessitent les données bas niveau pour tout calcul d'agrégat et de super-agrégat.

### 4.2.2 Sélections et projections successives

#### Sélections sur les cellules successives (*Dice*)

Nous montrons que, grâce à la propriété de commutativité des t-normes, le résultat de deux sélections successives sur les valeurs de cellules est indépendant de l'ordre d'application des opérations.

Soit $C_1''$ le cube obtenu en opérant une sélection sur les cellules du cube par le critère $\gamma_1$ suivi d'une sélection par le critère $\gamma_2$, et $C_2''$ le cube obtenu en appliquant le critère $\gamma_2$ suivi d'une sélection par le critère $\gamma_1$. Nous montrons que ces deux cubes sont équivalents,

ce qui revient à montrer que toutes les cellules contiennent les mêmes degrés $\mu$, ce degré étant le seul modifié par l'opération de sélection sur les valeurs de cellules.

Les t-normes étant associatives et commutatives, nous avons :

$\forall \overrightarrow{x}$,

$$\begin{aligned}\mu_{C_1''}(\overrightarrow{x}) &= \mathcal{T}(\mathcal{C}(v(\overrightarrow{x}),\mu_{\gamma_2}),d(\overrightarrow{x}),\mathcal{T}(\mathcal{C}(v(\overrightarrow{x}),\mu_{\gamma_1}),d(\overrightarrow{x}),\mu_C(\overrightarrow{x})))\\ &= \mathcal{T}(\mathcal{T}(\mathcal{C}(v(\overrightarrow{x}),\mu_{\gamma_2}),d(\overrightarrow{x}),\mu_C(\overrightarrow{x})),\mathcal{C}(v(\overrightarrow{x}),\mu_{\gamma_1}),d(\overrightarrow{x}))\\ &= \mathcal{T}(\mathcal{C}(v(\overrightarrow{x}),\mu_{\gamma_1}),d(\overrightarrow{x}),\mathcal{T}(\mathcal{C}(v(\overrightarrow{x}),\mu_{\gamma_2}),d(\overrightarrow{x}),\mu_C(\overrightarrow{x})))\\ &= \mu_{C_2''}(\overrightarrow{x})\end{aligned}$$

On remarque que, comme mentionné dans la section 2.3.3, le seul moyen d'obtenir le même cube en appliquant deux fois un même critère de sélection (même critère $\gamma$) est de considérer une t-norme idempotente.

Si pour chaque cellule $\overrightarrow{x}$, $\mathcal{T}(\mathcal{C}(v(\overrightarrow{x}),\mu_\gamma),d(\overrightarrow{x}),\mathcal{T}(\mathcal{C}(v(\overrightarrow{x}),\mu_\gamma),d(\overrightarrow{x}),\mu_C(\overrightarrow{x})))$ est égal à $\mathcal{T}(\mathcal{C}(v(\overrightarrow{x}),\mu_\gamma),d(\overrightarrow{x}),\mu_C(\overrightarrow{x}))$ alors $\mathcal{T}$ est nécessairement construit à l'aide de la t-norme min.

### Sélections successives sur les dimensions (*Slice*)

La démonstration de commutativité est très simple, et est similaire à la précédente pour l'opérateur *dice*. En opérant une sélection sur une même dimension par les critères $\gamma_1$ et $\gamma_2$, l'ordre n'a aucun effet sur le résultat. On note $C$ le cube de départ, $D$ la dimension sur laquelle s'applique la sélection, $dom(D)$ son domaine, $C_1''$ (resp. $C_2''$) le cube obtenu après sélection par le critère $\gamma_1$ (resp. $\gamma_2$) puis par le critère $\gamma_2$ (resp. $\gamma_1$). On a :

$\forall d_i \in dom(D)$,

$$\begin{aligned}d_{C_1''}(d_i) &= \top(\mathcal{C}(v(d_i),\mu_{\gamma_2}),\top(\mathcal{C}(v(d_i),\mu_{\gamma_1}),d_C(d_i)))\\ &= \top(\top(\mathcal{C}(v(d_i),\mu_{\gamma_2}),\mathcal{C}(v(d_i),\mu_{\gamma_1})),d_C(d_i)))\\ &= \top(\mathcal{C}(v(d_i),\mu_{\gamma_1}),\top(\mathcal{C}(v(d_i),\mu_{\gamma_2}),d_C(d_i)))\\ &= d_{C_2''}(d_i)\end{aligned}$$

On remarque que, comme indiqué dans la section 2.3.3, le seul moyen d'avoir le même cube résultat en opérant plusieurs fois une sélection par le même critère $\gamma$ est de considérer une t-norme idempotente.

Si pour chaque tranche $d_i \in dom(D)$, $\top(\mathcal{C}(v(d_i),\mu_\gamma),\top(\mathcal{C}(v(d_i),\mu_\gamma),d(d_i)))$ est égal à $\top(\mathcal{C}(v(d_i),\mu_\gamma),d(d_i))$ alors l'opérateur $\top$ est la t-norme min.

### Projections successives

On considère un ensemble de dimensions $\{A_1,...,A_n,B_1,...,B_m\}$ tel que $\{A_1,...,A_n\} \subseteq \{B_1,...,B_m\}$. Les deux requêtes (4.3) et (4.4) sont équivalentes, puisque rien n'est modifié par rapport au modèle d'Agrawal et al. [AGR 97].

$$Proj\ (Proj\ C\ B_1,...,B_m)\ A_1,...,A_n \tag{4.3}$$

$$Proj\ C\ A_1,...,A_n \tag{4.4}$$

En effet, la projection est l'opération consistant à détruire un ensemble de dimensions dont les domaines actifs sont réduits à des singletons. Dans ce cas, puisque l'ensemble des $A_i$ est inclus dans l'ensemble des $B_i$, détruire les $A_i$ détruit également les $B_i$.

**Conjonction de critères**

Nous étudions ici les équivalences entre des requêtes successives utilisant deux critères et une requête unique utilisant une conjonction de ces deux critères.

**Sélections sur les valeurs des cellules**

On note $\wedge$ l'opérateur de conjonction de critères.

Nous étudions ici si $SelCells\ (SelCells\ C\ \gamma_1)\ \gamma_2$ est equivalent à $SelCells\ C\ \gamma_1 \wedge \gamma_2$

On note $C_1''$ le cube obtenu en opérant une sélection sur les cellules du cube par le critère $\gamma_1$ puis par le critère $\gamma_2$, et $C_2''$ le cube obtenu en opérant une sélection sur les cellules par le critère $\gamma_1 \wedge \gamma_2$.

Pour tout $\overrightarrow{x}$, on a :

$$\mu_{C_1''}(\overrightarrow{x}) = \mathcal{T}(\mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_2}), d(\overrightarrow{x}), \mathcal{T}(\mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_1}), d(\overrightarrow{x}), \mu_C(\overrightarrow{x})))$$

et

$$\mu_{C_2''}(\overrightarrow{x}) = \mathcal{T}(\mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_1 \wedge \gamma_2}), d(\overrightarrow{x}), \mu_C(\overrightarrow{x}))$$

Un contre-exemple à l'égalité de ces deux valeurs peut être considéré. Dans le cas où la condition suivante est vraie, les équations ci-dessus ne sont pas équivalentes : (i) $\gamma_1 \wedge \gamma_2 = \emptyset$, (ii) $v \wedge \gamma_1 = \emptyset$, (iii) $v \wedge \gamma_2 = \emptyset$, (iv) la t-norme choisie pour agréger les degrés est *stricte*, c'est-à-dire telle que $T(a, b) > 0$ si $a > 0$ et $b > 0$. Remarquons que le *min* satisfait cette condition.

En fait dans ce cas, $C(v(\overrightarrow{x}), \mu_{\gamma_1})$ et $C(v(\overrightarrow{x}), \mu_{\gamma_2})$ seront différents de 0 tandis que $C(v(\overrightarrow{x}), \mu_{\gamma_1 \wedge \gamma_2})$ sera égal à 0.

Cependant, les sélections successives sur les cellules sont souvent faites pour raffiner un critère, et il n'y a aucun besoin d'opérer deux sélections successives avec les critères disjoints. Nous portons donc notre attention sur les cas où $\gamma_1 \wedge \gamma_2 \neq \emptyset$.

On considère des sélections successives pour le raffinement du critère de sélection, on a $\gamma_2 \subseteq \gamma_1$. Si la t-norme utilisée pour agréger les degrés n'est pas idempotente alors il existe des exemples où le résultat est différent en appliquant successivement les deux critères ou en appliquant simultanément $\gamma_1 \wedge \gamma_2 = \gamma_2$. Si la t-norme utilisée est idempotente alors on a :

$\mathcal{T}(\mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_2}), d(\overrightarrow{x}), \mathcal{T}(\mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_1}), d(\overrightarrow{x}), \mu_C(\overrightarrow{x})))$
$= \mathcal{T}(\top(\mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_2}), \mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_1})), d(\overrightarrow{x}), \mu_C(\overrightarrow{x}))$

Or $\mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_2}) \leq \mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_1})$ puisque $\gamma_2 \subseteq \gamma_1$ et que $\mathcal{C}$ est monotone (étant une mesure de satisfiabilité). Nous avons donc :

$$\begin{aligned} &\mathcal{T}(\top(\mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_2}), \mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_1})), d(\overrightarrow{x}), \mu_C(\overrightarrow{x})) \\ &= \mathcal{T}(\mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_2}), d(\overrightarrow{x}), \mu_C(\overrightarrow{x})) \\ &= \mathcal{T}(\mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_1 \wedge \gamma_2}), d(\overrightarrow{x}), \mu_C(\overrightarrow{x})) \end{aligned}$$

**Sélections sur les dimensions**

Le problème est le même que précédemment, pour vérifier si

$$SelDim\ (SelDim\ C\ D\ \gamma_1)\ D\ \gamma_2$$

est équivalent à

$$SelDim\ C\ D\ \gamma_1 \wedge \gamma 2$$

Cependant dans ce cas, des combinaisons peuvent être considérées. Dans ce cas, toutes les requêtes possibles sont alors équivalentes puisque chacune d'elle ne modifie qu'un seul degré de tranches, ce qui n'affecte en rien le cube résultat.

### 4.2.3 Combinaison d'opérateurs

Nous étudions ici les propriétés des combinaisons de projection, sélection sur les cellules et sur les tranches.

Une opération de sélection sur les dimensions (*slice*) ne peut être appliquée si la dimension a été préalablement détruite par une projection. En outre, si la projection et l'opération de sélection ne s'appliquent pas à la même dimension, alors l'ordre de calcul des deux opérations n'a aucune influence sur le cube résultat.

On note que la projection ne peut s'appliquer que si le domaine actif de la dimension à détruire ne contient qu'une seule valeur. Une opération de sélection ou de généralisation sur les valeurs de cette dimension peut donc être nécessaire.

Une opération de sélection sur les valeurs des cellules peut être combinée avec une opération de généralisation sans modification du résultat.

#### Combinaison de sélections sur les cellules et sur les dimensions

Ces deux opérations peuvent être appliquées successivement dans n'importe quel ordre sans influence sur le résultat puisqu'il n'y a pas d'interaction entre les degrés manipulés. En effet, l'opération de sélection sur les cellules ne modifie que les degrés $\mu$ d'appartenance des cellules au cube tandis que l'opération de sélection sur les dimensions modifie les degrés d'appartenance des tranches au cube.

#### Combinaison de généralisation et de projection

On note que, la plupart du temps, une opération de généralisation est nécessaire pour réduire le domaine actif d'une dimension à un singleton, et permettre ainsi l'application de l'opération de projection. Ainsi, si la dimension à généraliser est la même que celle à détruire, il est évident que la première opération doit être faite avant la seconde. On généralise ainsi la dimension jusqu'au niveau le plus élevé (souvent noté $ALL$).

Dans le cas où la généralisation ne porte pas sur la dimension à détruire, l'ordre n'a aucun effet sur le cube résultat.

#### Combinaison de généralisation et de sélection (*slice* ou *dice*)

Généraliser un cube modifie le domaine actif de la dimension sur laquelle s'effectue l'agrégation des données. Le contenu des cellules est également modifié (éléments et degrés $\mu$). Il faut donc être vigilant sur la compatibilité du critère de sélection en fonction des données. Par exemple, sélectionner des résultats de ventes par un critère décrivant ce que sont des ventes *faibles* n'a pas le même effet et la même définition selon le niveau de granularité auquel on se place. Le niveau de granularité dépend des opérations de généralisation appliquées précédemment. Si l'opération de généralisation appliquée est de type *comptage* ou *somme*, le critère de sélection n'est plus compatible avec les données une fois la généralisation effectuée.

## Troisième partie

# Analyse de données multidimensionnelles floues

Les systèmes de gestion de bases de données, et en particulier les systèmes de bases de données multidimensionnelles, placés au cœur des systèmes décisionnels, assurent la gestion efficace de gros volumes de données à des fins d'analyse. Les bases de données multidimensionnelles, initialement conçues pour les analyses OLAP, se sont révélées très efficaces pour la fouille de données. En effet, elles sont construites à des fins d'analyse et optimisent les calculs d'agrégats et le dénombrement des données, cruciaux dans le cadre de la fouille de données. Cependant, aucun système décisionnel complet multidimensionnel n'autorise la prise en compte de données imparfaitement connues ou l'application d'algorithmes de fouille de données floues. Cette partie est donc consacrée à l'intégration du modèle proposé à la fouille de données multidimensionnelles floues. Le système utilise les bases de données multidimensionnelles floues comme support pour la découverte de connaissances [LAU 00b, LAU 00a, LAU 01b, LAU 01c, LAU 01d, LAU 02a]. Les composants d'analyse sont intégrés dans une architecture modulaire permettant l'utilisation de divers outils de fouille de données. Le premier de ces modules est lié à l'analyse *prédictive* des données, par la construction d'arbres de décision flous, tandis que les deux autres modules sont des méthodes d'analyse *descriptive* des données, pour la construction de résumés flous et la découverte de cellules anormalement vides.

Cette partie est organisée de la manière suivante : le chapitre 1 est consacré à l'étude des systèmes existants dans la littérature et à la présentation générale du système proposé. Les chapitres suivants sont consacrés aux différents modules de fouille de données introduits : construction d'arbres de décision (chapitre 2), génération de résumés flous (chapitre 3), et découverte des cellules anormalement vides (chapitre 4).

# Chapitre 1

# Fouille de données multidimensionnelles en ligne

La fouille de données multidimensionnelles, ou *OLAP mining*, correspond à un processus de fouille de données (*data mining*) intégrant une composante OLAP, c'est-à-dire s'effectuant à partir de données multidimensionnelles. L'intérêt pour cette approche, bien que récent, est de plus en plus grand [HAN 97], [GUE 99], [MIN 99]. Ceci est dû aux avantages de la représentation multidimensionnelle en vue d'appliquer des algorithmes de data mining. En effet, les bases de données multidimensionnelles permettent d'une part d'obtenir de bons temps de réponse pour les calculs d'agrégats et les requêtes complexes et d'autre part de travailler sur les hiérarchies. La taille des bases est réduite par la construction de cubes de données à un niveau agrégé. La plupart des travaux sont issus de la communauté des bases de données et ont été menés sur la découverte de règles d'association [HAN 97], [KAM 97], [BOU 99b], [GUE 99].

Dans ce chapitre, nous présentons d'abord les systèmes existants puis introduisons l'architecture du système que nous proposons.

## 1.1 Systèmes de fouille de données multidimensionnelles

Le rôle des bases de données dans les processus de fouille de données n'est plus à démontrer [CHE 96, MAN 96, CHA 98]. Depuis les travaux sur les bases de données statistiques, les besoins d'outils efficaces de gestion de gros volumes de données se sont fait ressentir. Dans ce contexte, il est apparu que le contexte OLAP, et les bases de données multidimensionnelles qui le sous-tendent, sont particulièrement adaptés à l'analyse de gros volumes de données multidimensionnelles. Ces bases de données sont au centre des systèmes d'information dédiés aux applications décisionnelles.

[MIN 99] compare les différentes méthodes d'apprentissage susceptibles d'être couplées avec les bases multidimensionnelles. Les auteurs posent le problème de l'intégration des méthodes de construction d'arbres de décision et d'un système de gestion de bases de données mais ne proposent aucune solution concrète. Le principal système d'OLAP Mining est issu des travaux de J. Han sur DBMiner, qui est maintenant un logiciel commercialisé.

Le système `DBMiner` [HAN 97, HAN 98, TAM 98] est fondé sur une approche MOLAP, afin d'optimiser les performances en terme de temps de réponse, même si cette amélioration s'effectue au détriment de la taille des données susceptibles d'être gérées. Les données sont stockées dans des structures multidimensionnelles. Aucune opération supplémentaire

n'est nécessaire pour extraire des données de la base. L'utilisateur choisit les attributs qu'il veut voir apparaître dans le cube depuis la base relationnelle. Des hiérarchies sont spécifiées sur les dimensions de manière à ce que l'extraction de connaissances soit possible à différents niveaux d'agrégation. Plusieurs modules d'analyse sont proposés, fondés sur la caractérisation, la comparaison, l'association, la classification, la prédiction et le clustering.

Par exemple, la construction d'un arbre de décision est possible. De manière classique, quatre étapes sont alors suivies, consistant à (i) séparer la base d'apprentissage de la base de test, (ii) analyser la pertinence des attributs, (iii) construire l'arbre et (iv) le tester. Les opérations multidimensionnelles (*e.g. slice, dice*) sont utilisées pendant le processus de construction pour extraire pour chaque nœud le sous-ensemble de données associé et pour lui appliquer les fonctions de data mining.

Dans [GUE 99], les auteurs détaillent les aspects liés au contexte multidimensionnel qui peuvent aider dans le processus de fouille de données, en particulier pour la découverte de règles d'association. Les cubes matérialisés fournissent en effet directement, comme valeur des cellules, le support des motifs de taille maximale. Pour toute cellule dont la valeur est supérieure au seuil de fréquence considéré, on sait que tous les sous-cubes sont fréquents par la propriété bien connue d'anti-monotonie du support. Pour les cellules nulles, les sous-cubes ne sont pas calculés. Il est cependant possible qu'un sous-cube mène à un motif fréquent, celui-ci sera alors pris en compte à partir d'une autre cellule impliquant l'une des positions impliquées dans la position de la cellule vide. Pour les autres cellules où la valeur est inférieure au seuil minimal de support, on ne peut pas savoir si les sous-cubes sont fréquents ou non, il faut donc adopter des méthodes classiques en commençant par le calcul des motifs de taille 1. Cependant, des heuristiques permettent de choisir les cellules, et donc les dimensions ayant le plus de chances de mener à des motifs fréquents. L'algorithme CIF (*Cube Itemset Finder*) est proposé couplant les approches ascendante et descendante.

Cependant tous les systèmes existants sont limités pour représenter les données continues du monde réel. En effet, de telles données sont souvent discrétisées et introduites comme seuils de valeurs de tests sur leur univers de définition (par exemple, âge $\leq 18.2$). Cependant, de tels seuils sont difficilement explicables. La théorie des sous-ensembles flous doit donc être introduite pour intégrer du flou dans l'univers d'un ensemble de valeurs continues afin de mieux prendre en compte les données du monde réel, souvent incertaines et/ou imprécises.

## 1.2 Spécificités OLAP

Les bases de données multidimensionnelles constituent un contexte très particulier, dont les caractéristiques doivent être prises en compte avant tout processus d'analyse. En particulier, les bases de données multidimensionnelles se distinguent des bases classiques utilisées dans les processus d'apprentissage et de fouille de données sur les points suivants :

- multidimensionnalité,
- présence d'un attribut particulier : la mesure du cube,
- présence de hiérarchies.

### 1.2.1 Multidimensionnalité

La fouille de données n'est plus effectuée sur des tableaux binaires mais sur des données multidimensionnelles numériques.

Or la plupart des systèmes d'extraction de règles d'association sont dédiés à des contextes où une seule dimension est analysée, par exemple les produits dans des applications de type *panier de la ménagère.*

### 1.2.2 Mesures de cubes

Les mesures des cubes sont construites à partir de fonctions d'agrégation qui peuvent être classées en trois catégories [GRA 97] : *distributives* (*e.g. comptage*, *min*, *max*), *algébriques* (*e.g. moyenne*), ou *holistiques* (*e.g. mediane*).

Cette catégorisation montre que des optimisations de calcul peuvent être envisagées, afin d'éviter des calculs redondants quand des résultats déjà obtenus peuvent être utilisés (par exemple pour des généralisations successives, comme nous l'avons présenté précédemment).

Une autre catégorisation des fonctions d'agrégation est proposée par [PED 99b], qui distingue trois types d'opérateurs d'agrégation (voir page 41).

Les mesures forment une nouvelle dimension, ajoutée à celle du schéma (souvent relationnel) de l'entrepôt de données. Cette classification tend à montrer que l'on ne peut appliquer des fonctions d'agrégation sur les cellules des cubes sans que cela soit dangereux d'un point de vue sémantique, étant donné que les données ont déjà été agrégées.

De plus, il apparaît que l'on peut sans problème sommer les résultats de ventes de produits le long de la dimension temporelle (par exemple pour trouver le total des ventes sur l'année), le long des catégories de produits (pour retrouver le total des ventes toutes catégories confondues), ou le long de la dimension spatiale (pour retrouver le total des ventes pour tous les magasins). Cependant, il n'en est pas de même pour un cube représentant l'état des stocks. On peut alors sommer les volumes de stocks le long de la dimension spatiale (pour connaître l'état des stocks dans tous les entrepôts, le long des catégories de produits (pour connaître l'état du stock pour tous les produits), mais il n'est pas pertinent de sommer les volumes de stocks le long de la dimension temporelle. On parle alors de mesure *semi-additive* [NET 99].

### 1.2.3 Niveaux de granularité

L'exploitation des informations à différents niveaux de granularité est très intéressante et très particulière aux bases multidimensionnelles, les bases de données relationnelles n'étant pas conçues pour l'exploitation des hiérarchies à des fins d'analyse (ce qui nécessite des opérations de jointure très coûteuses).

Cependant, les informations doivent être maniées avec précaution en présence de hiérarchies.

D'une part, les informations très agrégées sont souvent très peu pertinentes et très peu informatives du fait de leur trop grande généralité. Ainsi, même si le processus d'agrégation réduit considérablement la taille de la base, le processus de fouille de données risque de ne produire que des résultats triviaux.

D'autre part, la description des données est dépendante du niveau de granularité. Par exemple, une fonction d'agrégation procédant par sommes des valeurs conduit à des descriptions de *ventes faibles* qui doivent absolument être estimées selon le niveau de granularité considéré, les nombres de ventes par ville ou par région n'étant pas du même ordre. Dans notre approche, nous précisons que pour la construction des arbres de décision flous, notre système estime la *proportion* de cellules répondant aux différents critères.

## 1.3 Architecture du système proposé

L'architecture du système proposé repose sur la coopération de plusieurs entités qui sont :

- un système de gestion de bases de données multidimensionnelles floues,
- une interface de requêtes OLAP,
- un ensemble de systèmes de fouille de données,
- une interface utilisateur.

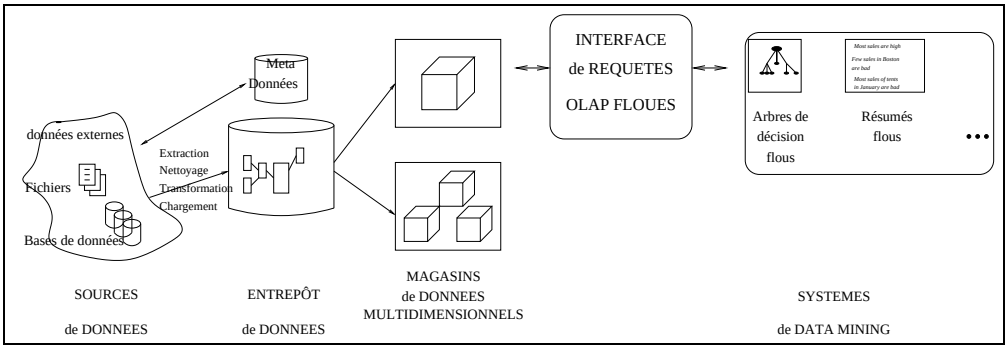


FIG. 27 – Architecture logicielle

Cette architecture est décrite pas la figure 27. Les différents systèmes d'apprentissage proposés dans le cadre de notre système sont détaillés ci-dessous. Ils sont installés de manière distribuée sur le réseau (voir figure 28) et fonctionnent sous divers systèmes d'exploitation. Cette architecture est transparente pour l'utilisateur, qui reste cependant en mesure de paramétrer le système en définissant les machines sur lesquelles s'exécutent les serveurs, et les ports de communication sur lesquels ils sont mis en écoute.

Trois modules de fouille de données sont disponibles dans notre système :

- construction d'arbres de décision flous,
- génération de résumés flous,
- recherche des cellules anormalement vides.

Toutefois, notre architecture est conçue pour rester modulaire et est donc ouverte à tout autre système.

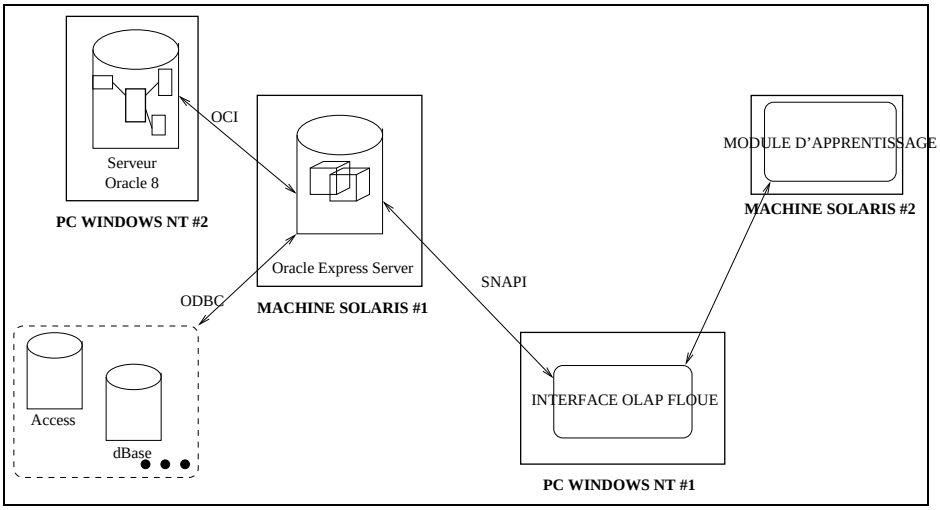


FIG. 28 – Architecture matérielle

# Chapitre 2

# Construction d'arbres de décision flous

Un arbre de décision est un cas particulier de questionnaire arborescent [PIC 65]. Il est construit à partir d'un ensemble d'exemples appelé base d'apprentissage. Chaque exemple de cette base est un cas résolu décrit par un couple [description, classe] où la description est un ensemble de couples [attribut, valeur] et la classe est un attribut particulier qui doit être expliqué ou reconnu. Il s'agit d'apprentissage supervisé. Un arbre de décision est une structure constituée de trois types d'éléments : les nœuds, les arcs et les feuilles.

Dans ce chapitre, nous présentons les arbres de décision flous puis introduisons le système mis en place pour la construction d'arbres de décision flous à partir de bases de données multidimensionnelles.

## 2.1 Arbres de décision flous

Dans un arbre de décision, chaque nœud est associé à une question sur une valeur d'attribut et chaque arc issu de ce nœud est associé à l'une des valeurs possibles de cet attribut, en réponse à la question posée. Chaque feuille de l'arbre est, elle, associée à une valeur de classe. Un arbre de décision est donc, d'une part, un outil de classification qui permet d'associer une valeur de classe à une description composée d'un ensemble de valeurs d'attribut, mais aussi, d'autre part, une représentation des connaissances.

La figure 29 présente un exemple d'arbre pour une application liée à une base de vente de produits sportifs. Sur cet exemple, la classe à déterminer est le taux de vente (*faible* ou *élevé*) en fonction du mois, du type de produit et de la ville. Le nœud racine de l'arbre est associé à une question sur la valeur de l'attribut *A2* (*type de produit*). Chaque arc issu de ce nœud est associé à une valeur de cet attribut. Les feuilles sont associées aux valeurs de la classe (*faible* ou *élevé*).

À un arbre de décision correspond une base de règles de production `si ... alors`. La prémisse d'une telle règle est composée des tests sur les valeurs des attributs associées à un chemin dans l'arbre de la racine vers une feuille, et la conclusion en est la valeur de la classe de la feuille du chemin. Dans l'exemple précédent (figure 29), on trouve, entre autres, la règle :

**Si** `A2 est 'RACQUETS' et A3 est 'DALLAS'` **alors** `'élevé'`

L'avantage principal des arbres de décision réside dans leur aspect *déclaratif*, c'est-à-dire qu'ils offrent une explication de la classification associée à une description (ce que des

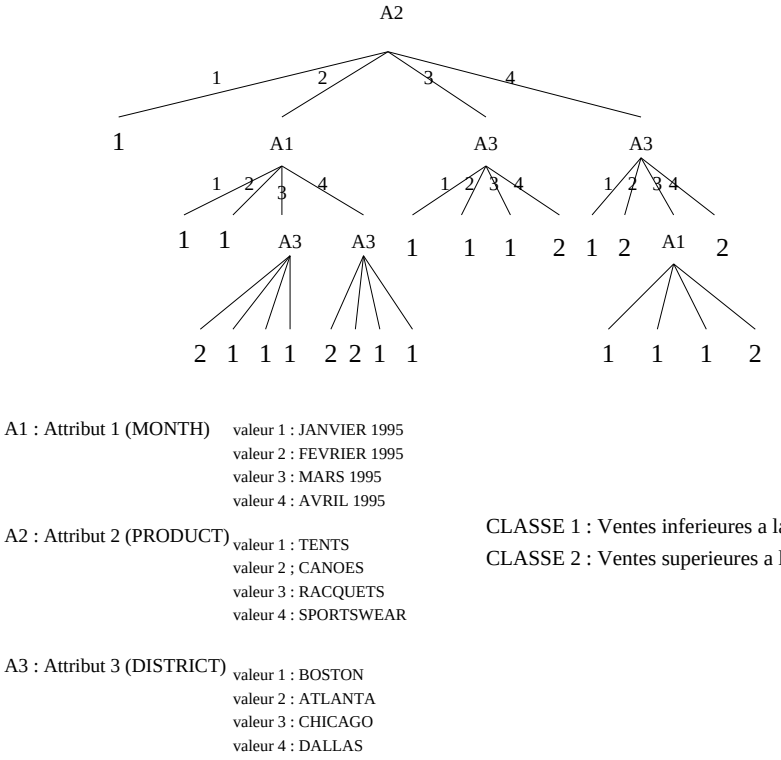


FIG. 29 – Exemple d'arbre de décision

modèles dits *procéduraux*, comme un réseau de neurones par exemple, ne permettent pas).

Comme toute méthode d'apprentissage inductif, un arbre de décision est construit à partir d'une base d'apprentissage. Il existe différents algorithmes pour construire un tel arbre à partir d'une base d'apprentissage, par exemple l'algorithme ID3 [QUI 86]. Ils procèdent tous sur un même modèle de construction de l'arbre depuis la racine jusqu'aux feuilles. Seuls certains paramètres utilisés pour les partitions successives de la base d'apprentissage changent [MAR 98b] : ces partitionnements s'effectuent grâce à une mesure de discrimination qui permet de choisir l'attribut partitionnant la base de manière optimale, c'est-à-dire l'attribut le plus pertinent dans la détermination de la valeur de la classe des éléments de la base d'apprentissage. L'entropie de Shannon est l'une de ces mesures, mais il en existe d'autres comme par exemple le test de Gini utilisé dans le système CART [BRE 84].

Si l'on considère une mesure de discrimination $H$, une stratégie de partitionnement $P$, un critère d'arrêt $T$, un ensemble d'apprentissage $\mathcal{E}$ composé d'exemples $e_i$ définis sur un ensemble d'attributs $A$ et une classe $C$, alors l'algorithme décrivant le processus de construction de l'arbre de décision peut être résumé de la façon suivante[12] [MAR 98b] :

[12] On note $e_i(A_j)$ la valeur pour l'exemple $e_i$ de l'attribut $A_j$.

1. En utilisant $H$, on choisit l'attribut $A_j \in A$ de modalités $\{v_{j_1}, ..., v_{j_{m_j}}\}$ qui partitionne au mieux l'ensemble d'apprentissage $\mathcal{E}$. Un nœud $\nu(A_j)$ est créé dans l'arbre, exécutant un test sur la valeur de $A_j$.

2. Suivant $P$, on partitionne l'ensemble d'apprentissage courant grâce à l'attribut $A_j$ en créant autant de sous-bases $\mathcal{E}_l$ que l'attribut a de modalités $v_{jl}$ : $\mathcal{E} = \bigcup \mathcal{E}_l$ avec $\mathcal{E}_l = \{e_i \in \mathcal{E} | e_i(A_j) = v_{jl}\}$.

3. En utilisant le critère $T$, on évalue le critère d'arrêt sur chaque sous-ensemble $\mathcal{E}_l$. Si le critère d'arrêt est vérifié alors une feuille est créée.

4. Retour à l'étape 1 pour tous les sous-ensembles qui ne vérifient pas le critère d'arrêt.

Dans l'algorithme ID3, la mesure d'entropie de Shannon (équation 2.1), issue de la théorie de l'information, est utilisée comme mesure de discrimination $H$.

$$H_S(C) = -\sum_{k=1}^{K} p(c_k) \log(p(c_k)) \tag{2.1}$$

où $\{c_k \mid k = 1, ..., K\}$ est l'ensemble des $K$ valeurs de la classe et $p(c_k)$ est la probabilité d'occurrence de la classe $c_k$ dans la base d'apprentissage.

Cette mesure est utilisée dans les travaux relatifs à la théorie des questionnaires sur la construction de questionnaires arborescents [PIC 65]. L'entropie de Shannon, en tant que mesure d'information, est utilisée pour sélectionner les questions afin d'obtenir le plus petit nombre possible de questions à poser pour déterminer une classe. Une bonne question fait baisser l'entropie pour les sous-ensembles résultants.

On utilise la mesure d'entropie conditionnelle ($H(C|A_j)$, *cf.* équation 2.2) pour mesurer le gain apporté en posant une question sur la valeur de l'attribut $A_j$ pour la reconnaissance de la classe de C. Si $H(C|A_j) = 0$ alors la connaissance de la valeur $v_{jl}$ pour $A_j$ détermine de façon unique la classe $c_k$ correspondante.

$$H(C|A_j) = -\sum_{l=1}^{m_j} p(v_{jl}) \sum_{k=1}^{K} p(c_k|v_{jl}) \log(p(c_k|v_{jl}) \tag{2.2}$$

où $\{c_k \mid k = 1, ..., K\}$ est l'ensemble des $K$ valeurs de la classe, $\{v_{jl} \mid l = 1, ..., m_j\}$ est l'ensemble des $m_j$ valeurs de l'attribut $A_j$, $p(v_{jl})$ est la probabilité d'occurrence de la valeur $v_{jl}$ dans la base d'apprentissage et $p(c_k|v_{jl})$ la probabilité conditionnelle de $c_k$ sachant $v_{jl}$ (occurrence de la classe $c_k$ pour les exemples ayant la valeur $v_{jl}$ dans la base d'apprentissage).

La mesure $H$ est souvent utilisée dans le critère d'arrêt $T$. Le but de l'algorithme est alors de poser autant de questions que nécessaire pour obtenir une valeur négligeable de l'entropie (*i.e.* inférieure à un seuil donné).

Les méthodes de construction d'arbres de décision les plus courantes traitent des valeurs symboliques possédant un nombre fini de valeurs ou des attributs numériques, prenant leurs valeurs dans un univers continu, après les avoir discrétisés en un ensemble fini de valeurs. Mais cette prise en compte des attributs numériques, ainsi que la prise en compte des attributs flous (par exemple environ 30 ans) peut être améliorée à l'aide de la théorie des sous-ensembles flous.

Dans un arbre de décision flou, les labels des arcs peuvent correspondre à des valeurs imprécises [BOU 97]. Les arbres de décision flous prennent en compte de telles valeurs soit durant leur construction, soit pendant la classification de nouveaux cas avec l'arbre. L'utilisation de la théorie des sous-ensembles flous améliore l'interprétabilité des arbres de décision dans le cas des attributs numériques. Un arbre de décision flou est équivalent à une base de règles floues : un chemin de l'arbre est alors équivalent à une règle floue, par exemple :

**Si** `la période est 'Début d'année' et la ville est 'dans les environs de Dallas'`
**Alors** `les ventes seront élevées.`

Un algorithme de construction d'arbres de décision flous est implanté dans le logiciel *Salammbô* [MAR 98b, MAR 99b]. L'arbre de décision flou construit par cette méthode à partir d'une base d'apprentissage peut alors être considéré comme un ensemble de règles de classification [BOU 99a].

## 2.2 Construction à partir de bases de données multidimensionnelles

Comme nous l'avons vu dans la section précédente, la construction d'arbres de décision flous, par l'utilisation de *Salammbô* par exemple, constitue un très bon moyen de traiter les données du monde réel et de construire des arbres de décision flous. Cependant, elle ne permet pas la prise en compte de gros volumes de données puisque les données sont chargées en mémoire pour être traitées. Or *Oracle Express* (OE) offre la possibilité de traiter un grand nombre de données provenant éventuellement de sources diverses comme c'est le cas dans les entrepôts de données. Nous étudions donc dans cette section l'interfaçage de ces deux composantes, l'une pour construire les arbres de décision flous, l'autre pour la gestion des données et les calculs d'agrégats [LAU 00a, LAU 00b].

Le problème principal est d'équilibrer le rôle de chaque composant (l'outil d'apprentissage flou et le système multidimensionnel) en ce qui concerne les calculs nécessaires. Dans leur version originelle, les outils d'apprentissage flou que nous intégrons effectuent eux-mêmes des agrégats et des calculs flous. Or, un système multidimensionnel est plus efficace pour exécuter ces calculs sur les données puisqu'il a été conçu afin de calculer de tels agrégats. Deux solutions sont alors possibles.

- Envoi d'agrégats simples sur les données (par exemple le nombre d'exemples correspondant à telle ou telle classe). L'intégration d'autres algorithmes d'apprentissage sera alors moins coûteuse. En effet, la plupart d'entre eux s'appuient déjà sur des statistiques de type comptage des données et gèrent eux-mêmes les traitements spécifiques de ces résultats. La solution est donc fonctionnelle pour divers algorithmes sans modification.
- Calcul d'agrégats complexes. Ces calculs peuvent inclure des opérations floues et des calculs d'agrégats et d'en transmettre le résultat. Les calculs d'agrégats seront alors gérés par le système de gestion de bases de données multidimensionnelles et les calculs d'agrégats partiels en cas de mise à jour des données seront optimisés. De plus, les échanges entre les deux systèmes seront réduits. Cependant, cette solution nécessite des modifications importantes au sein même du système de gestion de bases de données multidimensionnelles et se heurte à la pauvreté des syntaxes des langages proposées dans ce cadre.

Pour chaque étape dans la construction de l'arbre, un ensemble de données décrivant l'état courant est envoyé par *Salammbô* à l'application interface qui interroge alors la base gérée par OE. Ces données sont en fait des couples [attribut, valeur]. L'application interface interroge le cube contenant les données, calcule des opérations de comptage ou d'agrégats et renvoie les informations concernant les nombres d'exemples contenus dans le sous-ensemble pour chaque valeur d'attribut et pour chaque valeur de classe. Les sections 2.3 et 2.4 résument ces processus d'échanges.

## 2.3 Une solution de niveau élémentaire

Une première version a été étudiée pour laquelle les échanges entre composants concernent les statistiques associées aux sous-ensembles de données. L'algorithme de construction d'arbres présenté en 2.1 est donc adapté pour prendre en compte les échanges avec le serveur par l'intermédiaire de l'application interface.

L'étape 1 du processus décrit dans 2.1 est remplacée par le processus suivant :

> 1.1 *Salammbô* prépare une demande spécifiant la position courante dans l'arbre et l'envoie. La position courante est le chemin depuis la racine. Il est donné par une liste de type (attribut,valeur) où la valeur peut être soit un singleton, soit un ensemble de valeurs décrit par un intervalle.
>
> 1.2 L'application interface interroge le cube et renvoie pour chaque valeur d'attribut, pour chaque classe, les statistiques associées.
>
> 1.3 *Salammbô* reçoit les statistiques et effectue le calcul d'entropie.

En ce qui concerne le point 1.2., l'application interface extrait d'abord un sous-ensemble de données du cube et calcule ensuite des agrégats pour chacun de ces sous-ensembles afin de retrouver toutes les informations de comptage nécessaires à la construction de l'arbre.

> 1.2.a. Extraction (*slice and dice*) du sous-cube en limitant les modalités des dimensions aux valeurs transmises par *Salammbô*.
>
> 1.2.b. Pour chaque classe : *sélection*, grâce au critère d'appartenance à la classe, des cellules concernées et comptage des cellules non vides.

La figure 30 résume ce processus.

*Salammbô* calcule elle-même toutes les opérations nécessaires à la construction de l'arbre de décision flou en s'appuyant sur des échanges de statistiques relatifs à des intervalles. Des modalités sont alors construites et les valeurs continues sont discrétisées.

## 2.4 Une solution de niveau intégré

Afin d'améliorer les performances obtenues avec la première version, on exécute des calculs plus complexes au niveau du système de gestion de bases de données multidimensionnel. Une telle solution permet en effet de réduire les temps d'exécution ainsi que les temps consacrés aux échanges d'information entre les différentes composantes. En effet, les calculs exécutés par l'outil effectuant la tâche d'apprentissage sont essentiellement fondés sur des calculs d'agrégat, par exemple des agrégats de type comptage pour le calcul des probabilités utilisées dans le calcul de l'entropie de Shannon (voir équation 2.1). Ces calculs

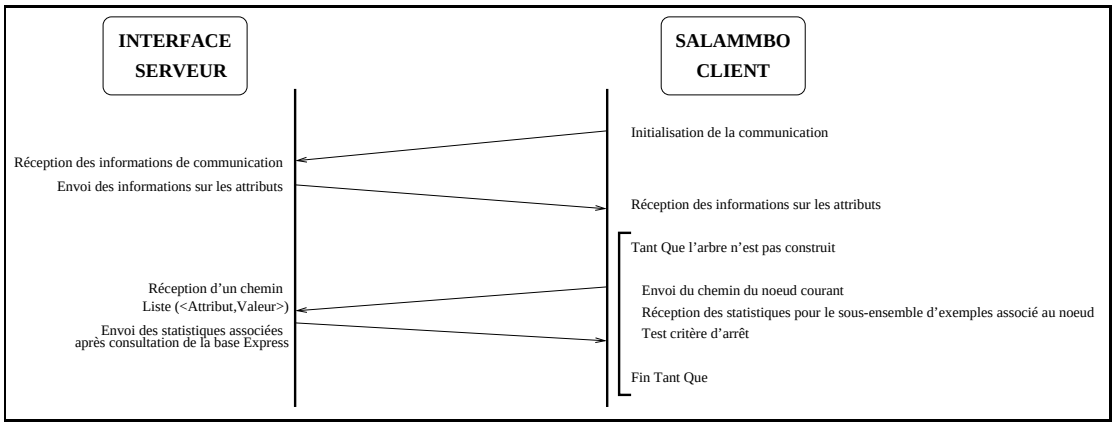


FIG. 30 – Construction de l'arbre : niveau élémentaire

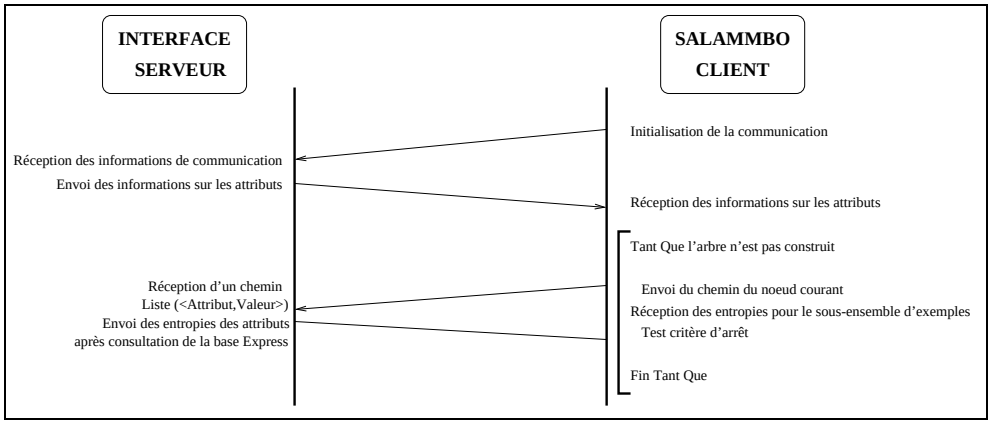


FIG. 31 – Construction de l'arbre : version intégrée

| | Approche de niveau | |
|---|---|---|
| | élémentaire | intégré |
| Intégration d'autres modules d'apprentissage | aisée | difficile |
| Limites syntaxiques | pas de limite | limites importantes des syntaxes des SGBD |
| Performances du système d'agrégation | non optimisées | optimisées par le SGBDM |

TAB. 3 – Comparaison des deux approches

sont donc optimisés par rapport à la solution précédente qui consiste à ne transmettre que les informations nécessaires sur les données et à laisser calculer la méthode d'apprentissage. Si l'on reprend l'exemple du calcul de l'entropie de Shannon, utilisé pour le choix de l'attribut ayant le meilleur pouvoir discriminant, alors on peut l'adapter de manière simple à l'environnement multidimensionnel et considérer le processus de construction adapté suivant :

> 1.1. *Salammbô* prépare une requête spécifiant la position courante dans l'arbre et l'envoie. Le chemin est donné par une liste de type (attribut,Valeur) où la valeur peut être soit un singleton, soit un ensemble de valeurs décrit par un intervalle.
>
> 1.2. L'application interface interroge le cube qui calcule les entropies pour chacune des dimensions du sous-cube spécifié par la requête et les renvoie.
>
> 1.3. *Salammbô* reçoit les mesures par attribut (qui vont lui permettre d'effectuer directement le partitionnement).

La figure 31 résume le processus de construction de l'arbre.

Le calcul de l'entropie peut s'effectuer grâce à l'opération *rollup* si le système prend en compte la définition de fonctions plus complexes que la somme ou grâce à des programmes internes, comme c'est le cas pour OE.

## 2.5 Comparaison des deux approches

Le tableau 3 résume les différents avantages et inconvénients des deux approches. On note que cette comparaison pour la construction des arbres de décision est également valable pour tout module susceptible d'être intégré à une base multidimensionnelle floue en vue de découvrir des connaissances pertinentes.

# Chapitre 3

# Génération de résumés flous

La production de résumés flous permet la génération de descriptions linguistiques des données. Les règles construites sont de la forme *Q des objets de la base qui sont A sont B*, où $Q$ est un quantificateur flou, et $A$ et $B$ sont des termes de résumé.

La littérature, abondante sur l'étude des quantificateurs réduits aux seuls cas existentiel et universel, est peu fournie en ce qui concerne l'étude sémantique des quantificateurs généralisés du type *la plupart*. Nous nous attachons donc dans un premier temps à décrire les travaux existants dans le domaine de la linguistique, puis à présenter les résultats d'une étude menée sur un ensemble de sujets pour la compréhension de l'usage des quantificateurs et de la production de résumés par les êtres humains.

Ensuite, la génération automatique de résumés flous est étudiée dans le cadre de notre système de fouille de données multidimensionnelles.

## 3.1 Quantificateurs et linguistique

La littérature linguistique est très peu développée en ce qui concerne l'étude des aspects sémantiques des quantificateurs flous du type *la plupart*. Les travaux sur les quantificateurs ont essentiellement concerné les quantificateurs existentiel et universel. En ce qui concerne les autres quantificateurs, les approches sont souvent syntaxiques et grammaticales, et concernent surtout la distinction des cas grammaticaux (utilisation des quantificateurs avec des noms pluriels ou singuliers, dénombrables ou non, etc.). Les principaux travaux axés sur la sémantique sont ceux de Georges Kleiber [KLE 87, KLE 90, KLE 94b].

[KLE 94b] présente quatre thèmes majeurs au travers de onze études, dont l'opposition entre massif et comptable. On appelle comptables les concepts du type *tabouret* que l'on peut dénombrer et massifs les concepts du type *tristesse*. Les noms comptables n'acceptent pas les déterminants discrets mais se combinent avec les quantificateurs *continus*.

Lorsque la quantification est exprimée par *assez, peu de, beaucoup de, pas mal de, etc*, la différence se manifeste dans la pluralisation, les comptables se présentant au pluriel, les non comptables au singulier (*beaucoup de tristesse, beaucoup de chaise*s). Certains quantificateurs, comme *un peu de* exigent des noms non comptables.

[KLE 94a] décrit l'emploi possible de l'article *le* pour décrire le cas général (*le castor construit des barrages*). [LIM 98] souligne lui aussi que cette tournure équivaut à un quantificateur quasi-universel, c'est-à-dire un quantificateur universel *affaibli* (on peut remplacer par *la plupart* ou *la majorité*). Dans ce même article, l'auteur étudie le double emploi de

l'adverbe *souvent*, soit comme marqueur de haute fréquence, soit comme quantificateur quasi-universel. De même que l'emploi de l'article *le*, les adverbes *généralement, habituellement* couvrent la quasi-totalité des objets ou individus considérés. L'adverbe *souvent* se réfère soit à la majorité des individus (les enfants sont *souvent* malins), soit à la majorité des occurrences, véhiculant ainsi l'idée de répétition (les manifestations sont *souvent* violentes). Les adverbes *quelquefois* et *rarement* sont distingués, le nombre d'occurrences se référant au premier étant petit mais non négligeable alors qu'il est négligeable dans le second cas.

Même si le contexte dans lequel nous nous plaçons n'est destiné qu'à des adultes, il est intéressant de constater que les enfants n'ont pas la même compréhension des quantificateurs que nous. Ainsi, il est apparu que les enfants interprètent mal les quantificateurs. Les enfants n'ont en effet pas la vision de ce qui n'est pas quand on leur présente ce qui est. Ainsi, [DER 98] montre que ce n'est qu'à partir de l'âge de 14 ans que l'utilisation du mot *certains* dans une phrase implique pour la plupart des sujets que *certains ... ne ... pas*, ceci n'étant que très peu observé chez les enfants de 10 ans (16% des sujets seulement). Dans le contexte de notre étude, ceci peut être très préjudiciable, cependant l'étude menée montre que 90% des adultes savent que quand il est dit que *certains* objets vérifient une propriété, alors *certains* objets *ne* la vérifient *pas*.

Concernant les aspects linguistiques de notre approche, nous ne visons pas une étude exhaustive menée d'un point de vue purement linguistique qui serait en dehors de notre propos, mais visons plutôt à étudier plus avant les relations entre les différents quantificateurs, notamment les relations de synonymie afin de conserver le moins de quantificateurs possible, de polysémie afin d'éviter toute confusion, et les relations d'implication (par exemple en reprenant l'exemple ci-dessus où un sujet adulte doit pouvoir induire que certains objets ne vérifient pas une propriété si certains autres la vérifient).

Dans le contexte dans lequel nous nous plaçons dans cette thèse, il est important de s'attacher aux aspects pragmatiques, c'est-à-dire non pas seulement liés à la signification des termes, mais enrichis par tout ce que la connaissance linguistique et contextuelle permet d'inférer. En effet, l'étude des quantificateurs est menée ici dans le contexte d'une application décisionnelle dédiée à des spécialistes du chaque champ d'application, où les connaissances des experts et les propriétés du domaine à analyser sont primordiales.

## 3.2 Étude expérimentale sur l'utilisation des quantificateurs

Dans cette section, nous détaillons les résultats de l'enquête effectuée auprès d'une centaine de personnes portant sur l'utilisation des quantificateurs flous dans les résumés de données numériques.

Une même série de résultats au baccalauréat est présentée deux fois aux sujets (voir annexe F). Dans la première phase, les sujets n'ont que la contrainte de ne pas répéter les pourcentages trouvés dans les tableaux de données. La seconde phase quant à elle impose le choix d'un ou plusieurs quantificateurs (préalablement définis) qui, selon le sujet, représentent au mieux les données de chacun des tableaux de résultats.

L'étude des résultats du test est menée selon trois perspectives :

- analyse par question : pour un même tableau de données numériques, analyse transversale des réponses sur l'ensemble des questionnaires,
- analyse par sujet : analyse des réponses d'un même sujet pour chaque questionnaire,

– analyse par quantificateur : analyse des occurrences et co-occurrences de chaque quantificateur utilisé.

Les sujets sont constitués en deux groupes principaux selon les conditions de passage du test. Le premier groupe (30 sujets) a répondu au questionnaire par internet, tandis que le second, constitué d'étudiants en psychologie de l'université Paris 8 (57 sujets), a répondu par écrit. La principale différence de condition de passage du test entre les deux groupes se situe au niveau du passage du deuxième test. Les sujets du second groupe avaient leurs réponses au premier questionnaire sous les yeux pendant qu'ils étaient soumis au deuxième questionnaire et celui-ci a été proposé immédiatement après le passage du premier questionnaire. Au contraire, les sujets du premier groupe n'ont répondu au deuxième questionnaire que quelques jours après le passage du premier test sans avoir le rappel de leurs premières réponses.

### 3.2.1 Questionnaire I : résumés produits sans contrainte

Les sujets ayant répondu au questionnaire, face à une question ouverte leur demandant de résumer les tableaux de données, ont adopté principalement deux stratégies. La première stratégie consiste à décrire la (ou les) modalité(s) dominante(s) et son impact à l'aide de *quantificateurs* flous, du type *la moitié*. Des modificateurs sont souvent employés, par exemple *très* .

La seconde stratégie est utilisée quand aucune modalité ne se détache. Elle consiste à **ordonnancer** les résultats, c'est-à-dire à produire des comparaisons (surtout pour la quasi-égalité) entre les différentes modalités présentées, par exemple en positionnant les résultats des filles par rapport à ceux des garçons, ce que l'on pourrait nommer *analyse relative* des résultats. Les termes utilisés sont alors du type *d'abord, ensuite* et aucune mesure ni quantitative ni qualitative n'est fournie pour évaluer la proportion d'objets mis en jeu. Dans le cas d'une variable intra-dimension, il est souvent possible de reconstruire les ordres de grandeur du tableau numérique à partir du résumé, la somme des proportions numériques étant de 100%. Cependant, dans le cas de variables inter-dimensions, il est impossible de retrouver l'ordre de grandeur.

Les termes utilisés sont principalement les suivants :
- Ordonnancement des valeurs numériques
  - plus de ... que de ... et moins de ... que de ...
  - d'abord ... ensuite/puis ... le reste ...
  - meilleur/mieux que ...
  - beaucoup ... peu
- Utilisation de quantificateurs
  - principalement, en général, la majorité, fréquemment, souvent, presque / quasiment tous, une bonne partie
  - (très) peu de, une minorité
  - un tiers, (près de) la moitié, plus d'1 million
  - de l'ordre de x%
- Relations (floues)
  - à peu près égal, (un peu) plus souvent, presque autant, pas de différence significative, baisse, augmente (pas forcément temporel), constant, plus grand, inférieur à, supérieur à
- Modificateurs flous

– quasiment, presque, légèrement, nettement, sensiblement, approximativement, (très) (faible) minorité, grande majorité

Les quantificateurs qui n'apparaissent pas aussi fréquemment que l'on aurait pu le penser sont les suivants :

– les approximations de proportions autres que *environ la moitié* (*environ un quart*, *environ un tiers*),
– *la plupart* n'est pas très utilisé de manière spontanée, par rapport à son utilisation quand il est proposé. Les sujets utilisent plutôt *beaucoup*,
– *plus* est très utilisé, les comparaisons et ordonnancements semblent être primordiaux

Quand une relation est établie (ordonnancement), on constate que les ordres de grandeur ne sont pas donnés. Typiquement, un seul sujet parmi les 57 du deuxième groupe a donné l'ordre de grandeur pour les questions 13 et 14, et tous ont ordonné les taux de réussite.

### 3.2.2 Questionnaire II : choix des quantificateurs

On analyse ici les résultats au questionnaire II.

Les choix des sujets pour résumer les données qui leur ont été présentées sont détaillés ci-dessous. Le nombre d'occurrences de chaque quantificateur est indiqué entre parenthèses. On rappelle que les sujets peuvent donner plusieurs réponses.

On construit le graphe des occurrences des principaux quantificateurs choisis par les sujets pour chaque question (dont la description est donnée en fin d'annexe F).

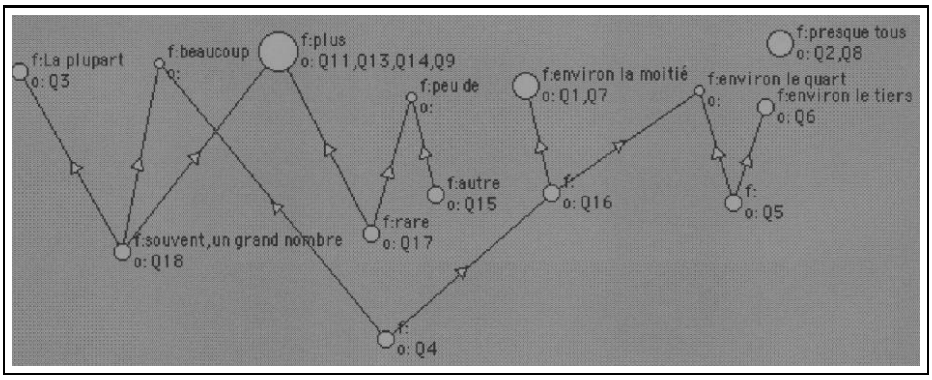


FIG. 32 – Graphe de choix - questionnaires internet

Si l'on ne considère que les résultats obtenus par internet, seules les questions 4 et 10 ont conduit à des réponses très réparties sur les quantificateurs proposés. Pour l'ensemble des autres questions, on construit le graphe d'occurrence des quantificateurs en représentant les quantificateurs apparaissant le plus. La figure 32 illustre les résultats hors question 10. On note qu'en ôtant la question 4, trois groupes de quantificateurs apparaissent tout à fait clairement, le premier constitué par le seul quantificateur *presque tous*, le deuxième par les quantificateurs liés aux proportions *environ la moitié, environ le quart, environ le tiers*, et le troisième constitué par les quantificateurs décrivant soit le cas général (*la plupart*), soit les cas particuliers (*peu de*).

La figure 33 illustre les résultats tous sujets confondus. On note que dans ce dernier cas, les sujets sont plus influencés par leurs réponses au premier questionnaire, d'une part parce

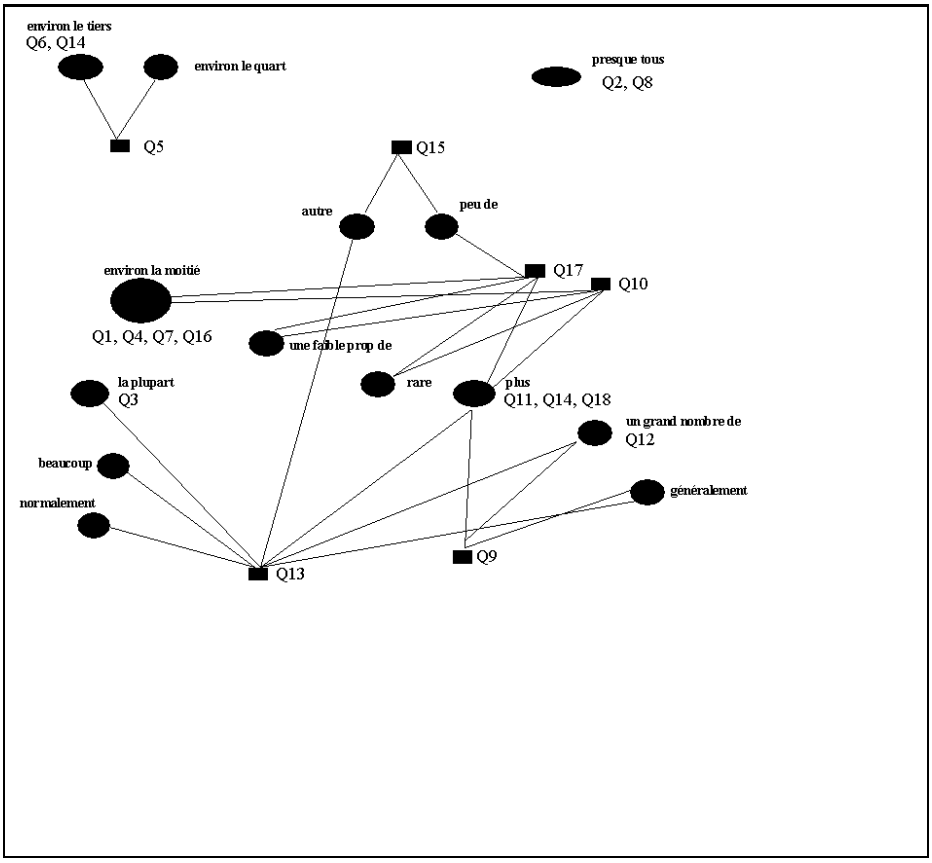


FIG. 33 – Graphe de choix tous questionnaires confondus

que la passation se fait juste après (alors que par internet, les contraintes de gestion de la réception et de l'envoi de l'adresse du second questionnaire ont imposé un délai de quelques jours), et d'autre part parce que les sujets avaient les réponses au premier questionnaire sous les yeux (contrairement à la passation par internet).

L'option *autre* est proposée pour les cas où le sujet pense qu'il manque le qualificatif compatible avec les données. Dans le questionnaire internet, les sujets n'étaient pas en mesure de spécifier quel autre quantificateur ils auraient souhaité utiliser. Cependant ce choix apparaît principalement en cas d'égalité des résultats pour remplacer les *autant que* et équivalents du premier questionnaire. L'étude des quantificateurs ajoutés à la liste dans le deuxième groupe de sujets montre que la même stratégie est mise en œuvre, les propositions étant les suivantes : *similaire, presque égale, plus de, ....*

Sur la question posée deux fois (valeurs numériques / proportion), faisant l'objet des questions (1) et (7), les mêmes quantificateurs sont utilisés, mais les résultats sont plus *éparpillés* quand la question n'est pas posée en utilisant des proportions.

Sur les questions (14) et (18) qui reprennent à peu près les mêmes données numériques mais sur des sujets différents (proportions fortes autour de 75%), on note que dans le premier questionnaire, la dimension temporelle est plus mise en valeur, par des termes du type *constant dans le temps, augmenté ...* et que les résultats sont assez homogènes pour le questionnaire 2.

### 3.2.3 Comparaison des réponses aux deux questionnaires

Les réponses à chacune des questions pour un même sujet sont souvent très proches pour les deux questionnaires, même si différents quantificateurs sont ajoutés alors qu'ils n'apparaissaient pas pour le premier questionnaire. Quand un ordonnancement était utilisé pour le premier questionnaire, les sujets se réfugient souvent en ajoutant un quantificateur à la liste, les sujets passant le test par internet choisissant l'option *autre*.

### 3.2.4 Test de la validité des résumés : questionnaire inverse

On se propose de valider les stratégies de construction de résumés par un troisième questionnaire. Celui-ci consiste à vérifier que les sujets associent les bonnes descriptions numériques aux résumés présentés. Ce questionnaire est décrit en annexe F.

116 sujets ont répondu à ce test. Les résultats montrent que les sujets réussissent à reconstruire les tableaux numériques à partir de résumés textuels. La dispersion des réponses autour de la moyenne est assez faible pour l'ensemble des réponses.

Les résultats des questionnaires sont pré-traités de manière à ne considérer que les sujets ayant répondu à la consigne. Des sous-ensembles flous sont construits à partir des distributions des résultats Plusieurs méthodes de construction de sous-ensembles flous sont possibles [ALA 97, BIL 96, BIL 99]. Dans notre cas, la méthode utilisée repose sur la prise en compte de l'histogramme normalisé des fréquences des réponses lissé en utilisant un filtre.

Les distributions sont décrites sur la figure 34. Les différences apparaissant sont liées aux questions auxquelles les réponses se réfèrent (voir questionnaire III en annexe F). La distribution associée au terme *majoritairement* est celle issue de la question 3. La distribution associée au terme *une majorité* est celle issue de la question 5. La distribution associée au terme *environ la moitié (1)* est celle issue de la question 4. La distribution

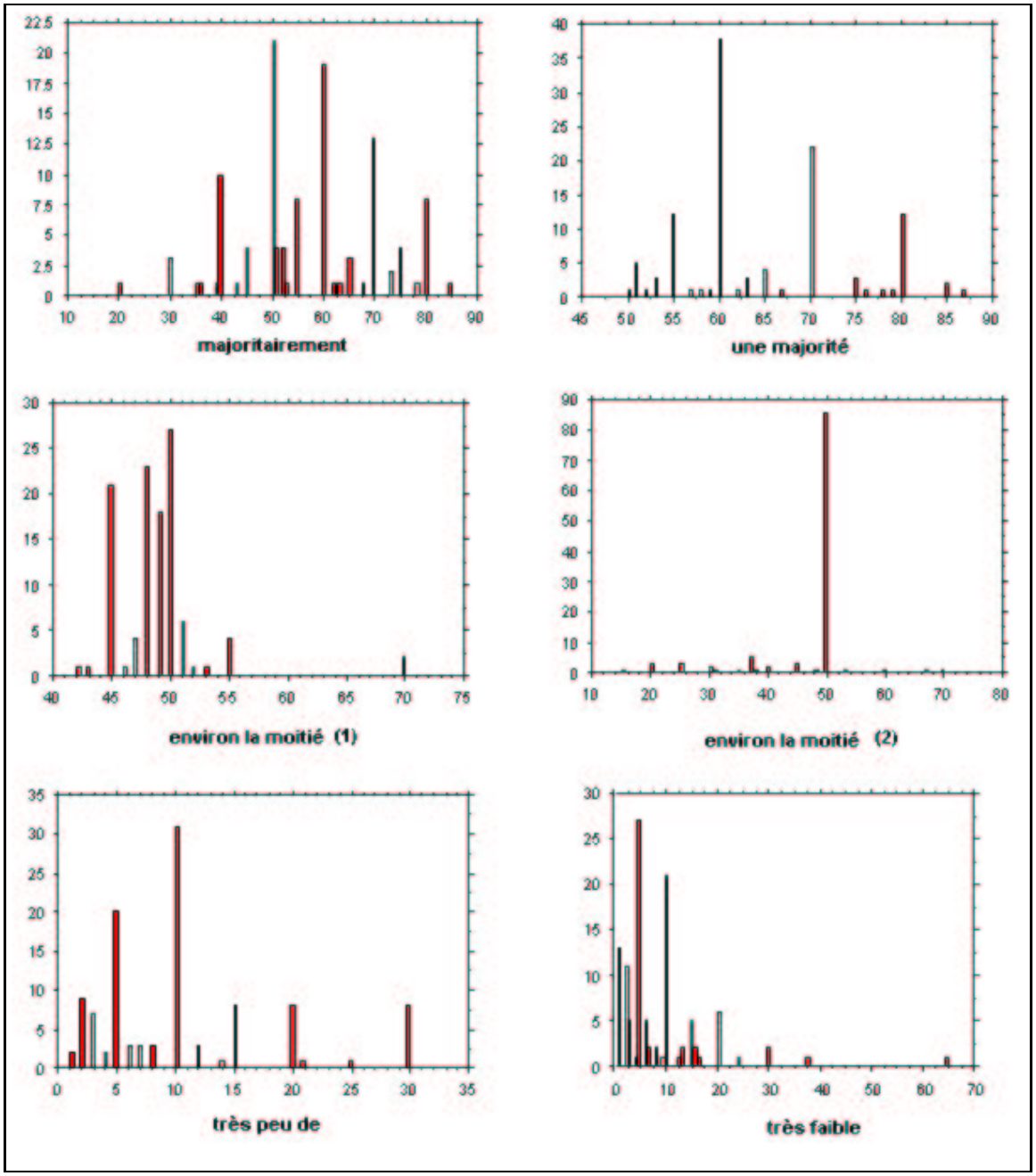


FIG. 34 – Distribution des réponses (l'abscisse représente le pourcentage et l'ordonnée le nombre de sujets ayant donné le pourcentage considéré)

associée au terme *environ la moitié (2)* est celle issue de la question 7. La distribution associée au terme *très peu de* est celle issue de la question 2. La distribution associée au terme *très faible* est celle issue de la question 8.

### 3.2.5 Conclusion

Pour conclure, nous notons que les tests effectués indiquent que les sujets, face à une situation les amenant à résumer des données numériques adoptent principalement les comportements suivants :

- utilisation du cas général (quantificateurs du type *la plupart*),
- utilisation de l'ordonnancement (termes du type *d'abord, ensuite, enfin*),
- et utilisation de la comparaison, surtout pour la quasi-égalité (termes du type *quasiment égal*).

Cette approche devrait être intégrée dans l'approche de génération des résumés flous, en plus de l'approche proposée dans les sections suivantes.

Cependant, si l'ordonnancement sur une tranche est très simple, le cas multidimensionnel pose problème. Il n'existe en effet pas forcément de manière d'arranger la mesure du cube en ordonnant les dimensions. Sur ces aspects, les travaux de [CHO 01] sont prometteurs.

En outre on note grâce au troisième test que les sujets sont en mesure de reconstituer les tableaux de données numériques à partir de résumés de la forme de ceux que notre système génère. Ce troisième test a de plus permis de construire les sous-ensembles flous correspondant aux quantificateurs relatifs les plus couramment utilisés et intégrés dans notre système, par exemple *peu de*.

## 3.3 Quantificateurs flous et résumés flous

### 3.3.1 Résumés flous : état de l'art

Les résumés flous ont été étudiés depuis le début des années 1980. Un tel résumé est par exemple donné par la phrase *"La plupart des ventes sont faibles"*. Ils ont un intérêt certain en découverte de connaissance puisque les données sont alors résumées en termes linguistiques, plus naturels pour les analystes.

Le concept de quantificateur généralisé a été introduit dans [MOS 57]. [Vää 97] passe en revue ces travaux et ceux qui les ont suivis. Par exemple, les notions telles que *la plupart* peuvent s'exprimer de la manière suivante :

$$Q_f(\ldots x \ldots) \Leftrightarrow \exists X(|X| \geq f(n) \wedge \forall x \in X(\ldots x \ldots))$$

avec $f : N \rightarrow N$ et $n$ la taille du modèle. On considère par exemple $f(n) = \lfloor n/2 \rfloor + 1$

Cependant cette définition utilise un seuil strict. Dans le contexte de la théorie des sous-ensembles flous, on appelle *quantificateur flou* un sous-ensemble flou représentant des quantités. Deux types de quantificateurs existent. Les premiers représentent des quantités absolues : *"Au moins 2, au plus 3" etc.*. Les seconds représentent des quantités relatives : *"environ la moitié, la plupart" etc.* Ces quantificateurs sont représentés par des sous-ensembles flous [Y.L 98a, Y.L 98b].

Soit $Q$ un quantificateur, $S$ un terme de résumé, $y$ la variable à résumer, et $\tau$ le degré de vérité du résumé, les résumés générés sont alors de la forme [KAC 00b, KAC 00a] :

"*Q y sont S : τ*"

ou "*Q B y sont S : τ*" où $B$ décrit par exemple l'importance (*"la plupart des experts importants sont jeunes"*).

On considère un ensemble de quantificateurs et de termes de résumés connus *a priori* (donnés par le système et/ou par l'utilisateur). Le système calcule le degré de vérité pour chacune des combinaisons possibles de $Q$ et $S$. Les quantificateurs et les termes de résumé sont des sous-ensembles flous. Les premiers sont définis sur l'intervalle $[0, 1]$ et les seconds sur l'intervalle de définition de $y$. L'introduction de sous-ensembles flous apporte plus de souplesse que des quantificateurs et termes classiques. Ils sont représentés par leurs fonctions d'appartenance ; $Q$ et $S$ sont ainsi respectivement représentés par $\mu_Q$ et $\mu_S$. On considère par exemple les quantificateurs *peu, environ la moitié, la plupart* illustrés par la figure 35.

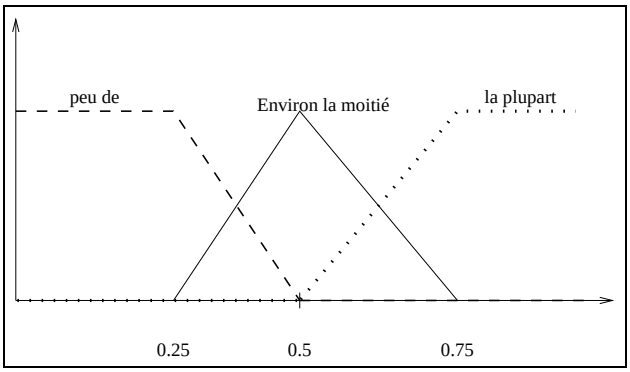


FIG. 35 – Exemples de quantificateurs

Le degré $\tau$ est alors calculé de la manière suivante :

$$\tau = \mu_Q \left( \frac{1}{n} \sum_{i=1}^{n} \mu_S(y_i) \right) \tag{3.1}$$

Dans les cas où un sous-ensemble de données est extrait correspondant aux données vérifiant le critère (flou ou non) $B$, on calcule ce degré de la manière suivante :

$$\tau = \mu_Q \left( \frac{1}{n} \sum_{i=1}^{n} \top(\mu_S(y_i), \mu_B(y_i)) \right) \tag{3.2}$$

où $\top$ est un opérateur de type t-norme.

Cinq mesures de qualité ont été définies, détaillées dans [KAC 00b] :
- La *valeur de vérité* ($T1$) correspond au degré $\tau$ présenté ci-dessus.
- Le *degré d'imprécision* ($T2$) renseigne sur le niveau de généralité du résumé.

$$T2 = 1 - \sqrt[m]{\prod_{j=1}^{m} (ins(s_j))}$$

avec $ins(s_j) = \frac{card\{x \in X_j : \mu_{s_j}(x) > 0\}}{card(X_j)}$

– Soit $w_g$ un critère flou de fonction d'appartenance $\mu_{w_g}$ portant sur l'attribut $V_g$. Le *degré de couverture* ($T3$) renseigne sur la proportion d'objets de la base concernés par le résumé.

$$T3 = \frac{\sum_{i=1}^{n} t_i}{\sum_{i=1}^{n} h_i}$$

avec $t_i = \begin{cases} 1 & \text{si } \mu_S(y_i) > 0 \text{ et } \mu_{w_g}(V_g(y_i)) > 0 \\ 0 & \text{sinon} \end{cases}$

et $h_i = \begin{cases} 1 & \text{si } \mu_{w_g}(V_g(y_i)) > 0 \\ 0 & \text{sinon} \end{cases}$

– Le *degré d'adéquation* (appropriateness) ($T4$) indique à quel point le résumé est caractéristique.

$$T4 = abs\left(\prod_{j=1}^{m} r_j - T3\right)$$

avec $r_j = \frac{1}{n}\left(\sum_{i=1}^{n} h_j\right)$ et $h_i = \begin{cases} 1 & \text{si } \mu_{S_j}(V_j(y_i)) > 0 \\ 0 & \text{sinon} \end{cases}$

– La *longueur du résumé* ($T5$) constitue la cinquième mesure de qualité.

$$T5 = 2 \times (0.5^{card(S)})$$

Ces mesures peuvent être agrégées afin de n'obtenir qu'un seul indicateur de la qualité du résumé. Les t-normes, ou les opérateurs de type OWA (opérateurs d'agrégation *O*rdered *W*eighted *A*veraging) introduits par Yager peuvent alors être utilisés [YAG 88].

Certains auteurs ont proposé d'autres approches pour résumer les données à l'aide de la théorie des sous-ensembles flous et de la théorie des possibilités. D'autres approches considèrent des données relationnelles et ont travaillé sur les dépendances fonctionnelles étendues au flou [BOS 98c], [CUB 97]. Dans ce cas, $X \rightarrow Y$ est valide si et seulement si pour tout couple de n-uplets de la relation, si les valeurs sur les attributs $X$ se ressemblent à un degré $\alpha$ alors les valeurs sur les attributs $Y$ se ressemblent à un niveau $\beta$.

Dans leurs approches, [BOS 00a, DUB 00], [RAS 00] considèrent également des bases de données relationnelles floues. La relation considérée est alors transformée en une nouvelle relation dont les valeurs d'attributs sont des étiquettes floues et où chaque ligne est un résumé. La relation est réécrite à différents niveaux de granularité. [RAS 00] génère des résumés de la forme *"Q des n-uplets sont $a_1$ et ... et $a_m$"* où les $a_i$ sont des étiquettes floues construites par le système en utilisant un algorithme de formation de concepts.

La génération de résumés flous sous forme de règles graduelles de la forme *"Plus les employés sont jeunes, plus leurs bonus sont faibles"* a également été étudiée [DUB 97b, BOS 99].

## 3.4 Résumés flous et règles d'association

Si les résumés flous n'ont pas été fortement utilisés dans la communauté des bases de données, les règles d'association, elles, dérivées des ensembles fréquents, sont très étudiées.

De nombreaux problèmes liés à l'extraction de règles suffisamment fréquentes et valides font l'objet de travaux [BOU 00a]. Des extensions aux règles d'association floues existent, afin de prendre en compte au mieux les attributs numériques en les partitionnant non pas à l'aide de seuils stricts, mais à l'aide d'intervalles flous.

Une règle d'association est une combinaison de valeurs d'attribut apparaissant en même temps avec une fréquence plus élevée que ce à quoi l'on pourrait s'attendre si les valeurs étaient indépendantes les unes des autres. Une règle d'association est constituée de deux parties, la partie *gauche*, ou *antécédent*, (notée ici $A$) et la partie *droite*, ou *conséquent*, (notée ici $B$), qui sont des ensembles de valeurs d'attibuts.

Plusieurs mesures existent pour décrire les règles d'association :

- La *couverture* : proportion d'exemples couverts par la partie gauche.
- Le *support* : proportion d'exemples couverts à la fois par les parties gauche et droite.
- La *confiance* : proportion d'exemples couverts par la partie gauche qui sont aussi couverts par la partie droite. Elle indique donc la probabilité qu'un cas satisfasse la partie droite s'il satisfait la partie gauche.
- Le *Lift* : résultat de la confiance divisée par la proportion d'exemples couverts par la partie droite. Il indique donc à quel point la partie droite est fréquente quand la partie gauche est plus fréquente que la normale.

Cependant nous nous focalisons sur le support et la confiance des règles. De nombreux algorithmes de découverte des règles d'association ont été proposés, à partir de la première définition de l'algorithme par Agrawal et al. [AGR 93].

D'une manière générale, ces algorithmes procèdent en deux phases :

1. génération des ensembles fréquents : sur une base contenant des exemples décrits selon $n$ attributs, les ensembles fréquents $\mathcal{L}_k$ pour $k = 1, .., n$ sont générés. Ce sont les ensembles de valeurs d'attributs apparaissant ensemble de manière fréquente, c'est-à-dire avec une proportion supérieure au support fixé. Les ensembles fréquents de taille $k$ sont générés à partir des ensembles fréquents de taille $k-1$, afin de limiter l'explosion combinatoire liée au nombre d'ensembles possibles.
2. génération des règles : Pour chaque ensemble fréquent découvert, on génère tous les couples (partie droite, partie gauche) possibles, et on génère les règles d'association pour tous les couples ayant une confiance supérieure au seuil fixé.

Nous présentons ci-dessous deux extensions des règles d'association allant dans le sens de nos travaux (qui eux couplent ces deux approches) : extension de la production de règles d'association à l'approche OLAP d'une part et au flou d'autre part.

### 3.4.1 Régles d'association et OLAP

La génération de règles d'association dans le contexte OLAP a été étudiée par certains auteurs, notamment dans le cadre des travaux de J. Han qui a introduit la problématique dans [HAN 98, KAM 97]. L'approche OLAP est très intéressante puisqu'elle permet la gestion de données pré-agrégées à différents niveaux possibles de granularité.

Les différences apportées par le contexte OLAP par rapport au contexte relationnel classique sont principalement les suivantes :

- les données à traiter ne sont pas des tableaux binaires mais des tables multidimensionnelles,
- les données sont agrégées (on ne dispose pas toujours des données sources),

– les dimensions sont organisées en hiérarchies.

Les règles multidimensionnelles, à l'opposé des règles *intra-dimensionnelles* explorent des relations entre variables. Ainsi, quand la seule dimension des *produits* est analysée dans le cadre du problème du *panier de la ménagère*, l'approche multidimensionnelle considère plusieurs attributs décrivant les données. Les auteurs ont également étudié les problématiques liées à la gestion de différents niveaux de granularité. Ces propositions ont été suivies dans [GUE 99, GUE 00]. Dans [ZHU 98], les règles d'association sont générées dans le contexte OLAP, la méthode ayant été intégrée dans DBMiner [HAN 96, HAN 98]. Dans cette approche, différents types de règles sont distingués : règles intra-dimensionnelles, inter-dimensionnelles et multi-niveaux.

Dans [BOU 99b], les auteurs s'attachent au processus de prétraitement et post-traitement des données en utilisant un langage de requête. Les données des cubes sont traduites en tables binaires sur lesquelles s'effectue la fouille de données. Cependant, aucune vision multi-niveau ne peut être produite.

[FEN 99] propose un système fournissant aux utilisateurs une vision en termes naturels de leurs données, comme nous l'avons présenté dans la section 3.4.1 de la page 41. Cependant, aucune automatisation de ce processus n'est fournie.

Dans nos travaux, nous nous attachons plus particulièrement aux règles inter-dimensionnelles, puisqu'elles sont plus liées d'une part à l'approche OLAP et d'autre part à l'approche des résumés flous [LU 00].

Une grande attention doit être apportée à la gestion de la mesure tant lors de la construction que lors de sa manipulation. Les méthodes de réduction des objets sur lesquels l'analyse portera changent en effet selon qu'une somme a été employée pour la construction des cubes et la navigation entre niveaux, ou que des comptages sont effectués. Ces problèmes ont été très peu étudiés dans la littérature du fait que les algorithmes existants se ramènent à des tables binaires. Dans le système que nous proposons, la mesure n'est pas simplement utilisée pour le comptage, mais constitue une dimension d'analyse en elle-même, après partitionnement des valeurs (numériques) des cellules obtenues par agrégation. Les concepts formant cette partition sont flous.

### 3.4.2 Régles d'association floues

Nous étudions dans cette section la différence entre règles d'association floues et résumés flous. Cette étude n'a malheureusement pas été conduite dans la littérature de manière avancée. [KAC 00b] propose d'utiliser l'approche des règles d'association pour la génération des résumés flous mais ne propose pas d'algorithme.

Dans l'approche par résumés flous, le support n'est pas considéré, au moins au moment de la construction des résumés, et constitue au mieux une mesure de qualité (voir $T3$), comme dans [KAC 00b]. Aucune méthode d'extraction *automatique* des résumés flous n'est proposée. La confiance est évaluée à travers le quantificateur flou utilisé dans le résumé, et un nouveau degré est introduit pour évaluer le degré de vérité du résumé (noté $\tau$ ou $T1$ précédemment).

Les principales différences entre les deux approches par règles d'association floues ou résumés flous sont décrites ci-dessous.

– Les règles d'association sont générées automatiquement tandis que les systèmes dédiés à la génération de résumés flous sont fondés sur l'évaluation de résumés constituant des hypothèses que l'utilisateur veut valider à l'aide d'un outil de requêtes

floues. On oppose donc, d'un côté un environnement qui procède par extraction automatique de toutes les règles à niveaux de support et confiance fixés, et d'un autre côté un système qui se contente de valider ou invalider une ou plusieurs hypothèses utilisateurs.
- Un quantificateur est utilisé pour exprimer la confiance dans le cas des résumés flous.
- Aucun seuil minimal de confiance n'est *a priori* considéré dans le cadre des résumés flous, et donc les quantificateurs du type *peu de* sont tout aussi utilisés que les quantificateurs exprimant le cas général, augmentant ainsi considérablement le nombre de résumés générés.
- La plupart des systèmes produisant des résumés flous n'exhibent pas le support de leurs règles, et surtout n'utilisent pas les propriétés de monotonie des ensembles fréquents pour réduire l'espace de recherche des résumés pertinents en utilisant un seuil minimal de couverture.
- Les règles d'association sont plutôt dédiées à l'analyse des relations entre valeurs d'un même attribut alors que les résumés flous sont orientés vers l'analyse multidimensionnelle.
- Les résumés flous sont présentés à l'utilisateur en utilisant des termes issus du langage naturel qu'il a même la plupart du temps défini lui-même.

Cependant, ces concepts sont très proches, par exemple les règles suivantes sont liées :

$$jeune \rightarrow mal\_payé \ (sup : 0.2, \ conf. : 0.8)$$
$$La \ plupart \ des \ jeunes \ sont \ dans \ la \ catégorie \ 'mal\_payé' \ (\tau = 1)$$

où *la plupart* est représenté par sa fonction d'appartenance $\mu_Q$ telle que $\mu_Q(0.8) = 1$.

Dans ces deux approches, une partition floue est nécessaire, ainsi qu'une méthode d'interrogation de la base de données à l'aide de critères flous et une méthode d'évaluation de cardinalités. Dans les deux cas, le nombre de jeunes ayant un revenu faible doit être évalué, dans le premier cas pour le calcul de la confiance de la règle et dans le second cas pour le calcul du degré de vérité.

La différence entre ces deux approches est donc très faible. La complexité et l'impossibilité de générer tous les résumés flous possibles vient du fait que cette solution reviendrait à générer l'ensemble des règles d'association à seuil de support et confiance nuls, avec en plus le calcul de tous les degrés de vérité pour tous les quantificateurs pris en compte.

## 3.5 Résumés flous de BDM floues

Dans cette section, nous proposons d'adapter le processus de génération des résumés flous aux bases de données multidimensionnelles [LAU 01c, LAU 03]. À notre connaissance, aucune étude n'a encore porté sur ce sujet. Nous présentons d'abord une première approche, en étudions les défauts, notamment dus à l'explosion combinatoire en complexité, et introduisons des méthodes de réduction du coût de la génération des résumés flous.

Dans cette section, les résumés sont du type *"Q des objets qui sont A sont B : $\tau$"* où $A$ est une combinaison de critères appelée *partie gauche* et $B$ est également une combinaison de critères constituant la *partie droite*.

### 3.5.1 Première approche

Dans l'approche proposée par [LAU 01b], les données sont potentiellement imprécises et incertaines. Elles sont alors plus ou moins susceptibles de participer au résumé selon

que leur adéquation à la fonction représentant le terme de résumé est plus ou moins forte. Le calcul de cette adéquation des données candidates au terme de résumé s'effectue en utilisant une mesure de comparaison qui va estimer à quel point la donnée (floue ou non) satisfait le terme utilisé dans le résumé.

Le calcul global du degré de vérité pour un résumé donné suit donc les étapes décrites par l'algorithme 1.

**Algorithme 1** Génération des résumés flous

1: Comptage des cellules non vides du cube
2: Sélection des cellules candidates
3: Comptage des cellules sélectionnées
4: Calcul du ratio entre les 2 comptages
5: Calcul du degré d'appartenance de ce ratio à la fonction représentant le quantificateur

Ceci revient à effectuer le calcul suivant :

$$\tau = \mu_Q \left( \frac{Comptage(cellules\ sélectionnées)}{Comptage(cellules)} \right)$$

Ce calcul est effectué pour chacun des quantificateurs, et l'utilisateur choisit de visualiser tous les résultats ou seulement le quantificateur pour lequel le degré de vérité est maximal.

Ces résumés sont améliorés pour permettre le calcul de la *couverture*, la génération de résumés plus complexes intégrant des termes de résumés et des pré-conditions constitués de conjonctions de termes et la prise en compte des hiérarchies.

### 3.5.2 Calcul du support et génération de résumés complexes

La notion de *couverture* n'est pas directement appréhendée par le résumé, dont le degré de vérité et le quantificateur reflètent plutôt la notion de *confiance*. Pour les résumés *simples* du type *"Q des objets sont B"*, support et confiance sont égaux. Pour les résumés comprenant une condition de présélection *"Q des objets qui sont A sont B"*, on peut considérer alors les degrés introduits par [KAC 00b] et notamment le degré de couverture $T3$. Pour un résumé du type *"Q des objets qui sont A sont B"*, le support serait donné par la proportion d'objets de la base qui sont à la fois $A$ et $B$ par rapport au nombre total d'objets. Le critère $A$ peut être un critère plus complexe constitué d'une conjonction de critères s'appliquant à différentes dimensions du cube. Si $A$ est vide, alors on revient au cas *simple*. L'utilisateur peut alors préciser quel support il désire de manière stricte (par exemple $support = 0.2$) ou de manière plus flexible, en précisant par exemple que le résumé doit couvrir *au moins un tiers* des exemples, où *au moins un tiers* est défini comme un sous-ensemble flou dont on connaît la fonction d'appartenance. Le processus de calcul du degré de vérité pour un résumé flou donné s'exprime alors comme une suite de requêtes OLAP flexibles :

#### Limites de l'algorithme

La génération de tous les résumés possibles est difficilement envisageable dès que le cube grossit en nombre de dimensions et que le nombre de termes de résumés et quantificateurs

**Algorithme 2** Génération des résumés flous avec calcul du support

1: {Le résumé à générer est de la forme}
2: {"Q des objets qui sont A sont B"}
3: $C_0 \leftarrow$ *Comptage des cellules non vides du cube*
4: Si $A = \emptyset$ alors $C_1 = C_0$ et aller à l'étape 7
5: Pré-sélection des cellules du cube par le critère $A$
6: $C_1 \leftarrow$ *Comptage des cellules du cube obtenu*
7: Sélection des cellules candidates pour le critère $B$
8: $C_2 \leftarrow$ *Comptage des cellules sélectionnées*
9: $Support \leftarrow C_2/C_0$
10: $Confiance \leftarrow C_2/C_1$
11: Calcul du degré d'appartenance de la confiance à la fonction représentant le quantificateur

disponibles augmente.
On considère $q$ quantificateurs et un cube $C$ ayant $k$ dimensions (y compris la mesure), avec sur chacune de ses $k$ dimensions $c_k$ termes de résumés possibles. En générant des résumés du type *"Q des y sont S"*, il existe $q \cdot c_k$ résumés possibles pour chacune des $k$ dimensions, soit $\sum_{j=1}^{k} q \cdot c_j$ résumés au total. Considérons maintenant des résumés plus complexes dans lesquels les données peuvent être pré-qualifiées : *"Q des y qui sont B sont S"* où la qualification supplémentaire $B$ est exprimée en fonction de critères sur les $k-1$ dimensions restantes. La qualification des objets se fera alors à l'aide de $i$ critères avec $i \in [0, k-1]$. Le nombre de combinaisons possibles pour exprimer $B$ est alors fonction du nombre de critères disponibles sur chacune des $i$ dimensions choisies, soit $\prod_{l=0}^{i} c_l$. Etant donné que $i$ varie entre 0 et $k-1$, on a donc $\sum_{i=0}^{k-1} \prod_{l=0}^{i} c_l$ manières de qualifier les objets, une fois la dimension présente dans la partie droite choisie. Le nombre de résumés possibles est alors :

$$q \sum_{j=1}^{k} \left( c_j \sum_{i \neq j} \left( \prod_{l=0}^{i} c_l \right) \right)$$

On cherche donc des méthodes de réduction du nombre de résumés générés. Ces méthodes sont présentées dans la section 3.7.

## 3.6 Prise en compte des hiérarchies

Le modèle multidimensionnel est adapté au traitement des données à différents niveaux de granularité. Les hiérarchies guident la navigation à travers ces niveaux. Dans leur approche, [RAS 00] génèrent une hiérarchie de concepts flous. Dans notre approche, la discrétisation des attributs numériques a notamment été utilisée pour la construction

d'arbres de décisions flous et pourrait guider une éventuelle construction de hiérarchies. Cependant, on considère ici des hiérarchies connues permettant aux objets d'appartenir graduellement à une ou plusieurs valeurs d'un niveau de granularité supérieur.

**Résumés à granularités différentes**

Si l'on ajoute aux possibilités de résumé tous les niveaux possibles, y compris les possibilités de croiser plusieurs niveaux sur les différents attributs, le nombre de résumés à générer est alors trop important.

Le cube est donc traité une première fois à un niveau de granularité élevé, et l'utilisateur peut ensuite sélectionner les résumés générés qu'il veut affiner. Cependant, le traitement à un niveau élevé risque de produire des résumés sans véritable valeur informative, comme le souligne [KAC 00b] pour le degré $T2$ d'imprécision.

Ceci rejoint les problématiques soulevées par [KAM 97] pour la découverte de règles d'association, puisque les supports élevés se trouveront à des niveaux de granularité élevés alors que les règles intéressantes sont plutôt susceptibles de se trouver à des niveaux de granularité plus faibles, avec des supports faibles, et ne seront donc pas découvertes.

Notre modèle autorise la découverte de résumés à des niveaux de granularité différents, la généralisation par comptage étant possible. Dans cette approche, on calcule un cube généralisé par comptage. Celui-ci peut être pré-calculé pour réduire les temps de réponse face aux requêtes utilisateur. Les différents résumés sont générés à ce niveau, cette opération sera d'autant moins coûteuse que le niveau de hiérarchie est très général. L'utilisateur choisit ensuite d'explorer plus avant l'un ou l'autre des pans du cube en sélectionnant un résumé et en indiquant les dimensions à regarder de manière plus détaillée. Si dans une hiérarchie classique on doit considérer alors les valeurs qui se généralisent dans l'ancienne, on doit cette fois considérer les valeurs dont le coefficient de passage vers l'ancienne valeur est supérieur à 0. Les résumés sont ensuite générés à nouveau pour ce nouveau niveau de granularité. L'utilisateur peut choisir de spécifier plusieurs résumés.

**Résumés intra-dimensions**

On nomme *résumés intra-dimensions* des résumés du type *"La plupart des ventes moyennes effectuées dans les villes de l'Est ont eu lieu à Chicago"*, où le concept de vente *moyenne* est représenté par un sous-ensemble flou de l'univers des ventes possibles. Il s'agit donc de résumés où la pré-sélection s'effectue sur les valeurs de hiérarchie qui se trouvent à des niveaux de granularité élevés. Le modèle que nous avons introduit est adapté à la génération de tels résumés puisqu'il répond d'une part aux besoins de navigation entre les différents niveaux de hiérarchie, et qu'il autorise d'autre part l'application de requêtes flexibles. On introduit un autre type de sélection qui concerne la sélection par rapport à la hiérarchie. Celui-ci est très proche de la sélection sur les valeurs de dimensions, où le calcul des nouveaux degrés des tranches aux cube est quelque peu modifié :

$$d_{C'}(d_i) = \top(c(d_i, d'_i), d'(d_i))$$

où $d'_i$ est la valeur du niveau plus élevé vers laquelle on effectue la généralisation et $c(d'_i, d_i)$ est le coefficient reflétant le degré avec lequel la valeur $d_i$ du cube de départ se généralise en la valeur $d'_i$ du cube d'arrivée. Par exemple, on aura $c(CHICAGO, EST) = 0.9$ calculé à partir de la fermeture transitive de la relation d'ordre floue caractérisant la hiérarchie spatiale.

De même que précédemment, le nombre de résumés que l'on peut potentiellement générer est très important, et des moyens pour réduire les calculs sont nécessaires. On

considère les résumés dont la partie gauche est constituée par une combinaison de termes relatifs à une même dimension à des niveaux de granularité différents et la partie droite par un critère décrivant la valeur de la mesure. Les éléments constituant la partie gauche du résumé doivent être compatibles. Dans le cas d'une hiérarchie définie par une relation d'ordre floue $R$, de fermeture transitive $R_T$ on dit qu'un élément $a$ est compatible avec un autre élément $b$ si $f_{R_T}(a, b) > 0$.
L'algorithme 3 décrit le processus de génération naïve de tels résumés.

**Algorithme 3** Génération des résumés flous multi-niveaux

1: **Pour chaque** dimension **Faire**
2: **Pour chaque** élément $e_s$ du niveau supérieur **Faire**
3: Trouver les éléments $e_i$ du niveau inférieur compatibles avec $e_s$
4: **Pour chaque** valeur $c$ qualifiant la mesure **Faire**
5: Trouver le quantificateur $Q$ le meilleur et générer le résumé
"$Q$ des objets qui sont $c$ concernant les $e_s$ sont $e_i$"
6: **Fin Pour**
7: **Fin Pour**
8: **Fin Pour**

On considère $q$ quantificateurs et $c$ termes de résumé possibles pour décrire la mesure. Pour une dimension donnée $D_i$, on note $s_k$ l'ensemble des éléments des niveaux de granularité supérieurs et $i(s)$ $(s \in s_k)$ l'ensemble des éléments du niveau inférieur compatibles avec la valeur $s$. On note $|i(s)|$ le nombre de ces éléments. Le nombre de résumés potentiels est alors :

$$q \cdot c \cdot \prod_{s \in s_k} |i(s)|$$

De même que précédemment, les limites de ces approches se font vite ressentir et la génération de tous les résumés possibles devient impossible car trop coûteuse. On introduit donc les algorithmes de génération de règles d'association et quelques méthodes de réduction des calculs.

## 3.7 Réduction du coût de la génération des résumés flous

Le coût de la génération des résumés est dû aux nombreuses combinaisons possibles entre les termes de résumés possibles. Le test de tous les résumés possibles devient impossible, le nombre de requêtes de sélection et de comptages coûteux étant trop important.

Différentes méthodes de réduction du coût de génération des résumés sont envisagées, reposant sur :

- les choix utilisateur,
- l'utilisation des algorithmes de génération des motifs fréquents,
- la mise à profit des propriétés sur les termes de résumé et sur les quantificateurs, et
- les spécificités du modèle multidimensionnel.

### 3.7.1 Utilisateur

Les attentes de l'utilisateur doivent être considérées. En effet, en fonction de celles-ci on peut délimiter le champ des quantificateurs et surtout des critères à utiliser. Ainsi, on présentera des quantificateurs relatifs aux faibles proportions (*très peu*) pour les utilisateurs s'intéressant aux cas particuliers et les quantificateurs sensibles aux fortes proportions (*la plupart*) pour les utilisateurs désireux de caractériser le cas général. La réduction du nombre de quantificateurs est moins cruciale que la réduction du nombre de termes susceptibles d'intervenir dans le résumé. En effet, une fois les sélections et les comptages effectués, la comparaison du degré obtenu avec les fonctions d'appartenance caractérisant les quantificateurs est aisée. En revanche, si l'utilisateur décrit ses attentes en réduisant le champ de recherche des combinaisons de termes de résumé possibles, le coût de la génération est d'autant plus réduit. Par exemple, l'utilisateur qui choisit de ne considérer que des résumés *simples* (on nomme *simple* un résumé du type *"Q des objets sont S"* par opposition aux résumés dont les objets sont pré-qualifiés *"Q des objets qui sont B sont S"*) ou qui choisit de ne considérer que $l$ dimensions parmi les $k$ dimensions disponibles ($l \ll k$) réduit considérablement la tâche du système. Par exemple, un utilisateur peut choisir de ne considérer que les résumés dont le terme de résumé $S$ correspond à un critère qualifiant la mesure du cube considéré. Dans le contexte des bases de données multidimensionnelles, ce choix s'avère souvent efficace par rapport aux attentes de l'utilisateur car les cubes sont construits dans un but d'analyse déterminé pour lequel la mesure revêt une importance particulière.

L'intervention de l'utilisateur dans le processus de génération des résumés ou de règles d'association est utilisée dans de nombreux systèmes. Dans [KAM 97], la génération des règles d'association est guidée par des méta-règles définies par l'utilisateur. De même, [KAC 00b] proposent de réduire l'espace de recherche des résumés flous en guidant la génération par les choix de l'utilisateur qui vise plutôt à valider ou invalider des hypothèses qu'à générer automatiquement tous les résumés valides. Celui-ci choisit les attributs sur lesquels travailler, ainsi que les valeurs à considérer. Les résumés générés dans [BOS 00a] sont également guidés par l'utilisateur qui choisit le type de résumé à produire et spécifie par exemple les attributs ou les relations entre attributs qu'il veut mettre en évidence. Dans notre système, l'utilisateur est en mesure de choisir :

- les dimensions à prendre en compte dans le résumé,
- les termes de résumés susceptibles de l'intéresser,
- la couverture souhaitée des résumés produits,
- les quantificateurs à considérer,
- les niveaux de granularité souhaités,
- les résumés et dimensions à détailler.

### 3.7.2 Génération des ensembles fréquents

La découverte de règles d'association est beaucoup utilisée et de nombreux algorithmes ont été étudiés pour réduire le coût de la génération de ces règles. L'application de ces méthodes aux cubes a été étudiée dans [ZHU 98]. De plus, les travaux concernant la génération des résumés flous [KAC 00b] adoptent également les algorithmes de génération de règles d'association pour réduire le coût de la génération des résumés.

A l'étape 1., il s'agit simplement d'éliminer les tranches du cube correspondant à des données non fréquentes. Cependant, les étapes suivantes ne s'expriment pas de manière

**Algorithme 4** Génération des résumés flous avec calcul des ensembles fréquents

1: Génération des ensembles fréquents de taille 1

2: **Pour chaque** $k$ à partir de $k = 2$ et tant que l'ensemble fréquent de taille $k-1$ est non vide **Faire**

3: Génération des ensembles candidats de taille $k$ en fonction des ensembles fréquents de taille $k-1$. Deux ensembles fréquents forment un candidat s'ils partagent $k-2$ termes identiques. L'ensemble candidat est alors formé des $k-2$ termes identiques et de chacun des termes distincts provenant des deux ensembles fréquents différents.

4: Pour chaque ensemble candidat : Sélection et comptage puis ajout des ensembles candidats de support satisfaisant aux ensembles fréquents de taille $k$

5: **Fin Pour**

6: Génération des résumés (algorithme 4.5)

aussi simple pour les motifs fréquents de taille supérieure à 1.

Les résumés flous sont proches des règles d'association. Le lien a déjà été fait par [KAC 00b] et les algorithmes d'extraction de règles d'association peuvent alors être utilisés pour optimiser la recherche des résumés flous potentiellement intéressants. Les résumés flous sont alors découverts en suivant l'algorithme classique de recherche des ensembles fréquents puis extraction des règles de confiance suffisante (voir algorithmes 4 et 5).

Les calculs de degré de vérité sont issus des étapes précédentes si toutes les valeurs de comptages sont conservées, et ne nécessitent aucun calcul supplémentaire. En effet, les sélections et les comptages correspondant à tous les ensembles fréquents ont déjà été effectués et il ne reste qu'à calculer les différents ratios. Cet algorithme réduit considérablement le coût de la génération des résumés flous. Le support minimal est spécifié par l'utilisateur (dans cette approche, les termes *couverture* et *support* se rapportent au même concept).

La génération des résumés candidats à partir des ensembles fréquents suit le processus décrit par l'algorithme 4.5.

La production des motifs fréquents est coûteuse et de nombreux travaux ont été conduits sur ce problème, considérant la phase de génération des règles à partir des fréquents comme très simple ([BOU 00d], [BOU 00b], [BOU 00c], [BAS 02]). Cependant, la génération de tous les résumés (ou toutes les règles) est assez coûteuse si l'on recherche les résumés ayant une partie droite de taille quelconque. Pour chaque motif fréquent de taille $k$ généré, $2^k - 1$ résumés sont possibles (on ne considère pas le résumé à partie droite vide). Pour ce qui concerne les règles d'association classiques, il est possible de réduire le coût de génération des règles en considérant les sous-ensembles des fréquents de manière récursive. Par exemple pour un motif fréquent $ABCD$, on considère d'abord $ABC$ puis $AB$ etc. La propriété de confiance devient alors une propriété de coupure car si $ABC \Rightarrow D$ n'a pas une confiance suffisante, alors il n'est plus besoin de considérer la règle $AB \Rightarrow CD$[13].

Dans le cas de résumés flous, cette propriété n'est valide que si l'on considère uniquement des quantificateurs flous dont la fonction d'appartenance est monotone croissante sur $[0, 1]$ (par exemple, *la plupart*). Le degré de vérité se comporte alors comme le degré de confiance et constitue une contrainte permettant d'effectuer des coupures lors de la phase

[13]En effet, $support(ABC) \leq sup(AB)$ donc $\frac{support(ABCD)}{support(ABC)} \geq \frac{support(ABCD)}{support(AB)}$, c'est-à-dire $conf(ABC \Rightarrow D) \geq conf(AB \Rightarrow CD)$

de génération des règles. Une autre propriété est étudiée dans la section suivante.

**Algorithme 5** Génération des résumés à partir des ensembles fréquents

1: $C\_Resumes \leftarrow \emptyset$ {Ensemble des résumés candidats}
2: {On génère l'ensemble des résumés candidats}
3: **Pour chaque** ensemble fréquent **Faire**
4: $C\_Part \leftarrow$ ensemble des couples de partitions de taille 2 de l'ensemble fréquent
5: $C\_Resumes \leftarrow C\_Resumes \bigcup C\_Part$
6: **Fin Pour**
7: **Pour chaque** résumé candidat $(A, B)$ **Faire**
8: Recherche des comptages par sélection sur $A$, sur $B$ et sur $A \wedge B$
9: {Tous les comptages ont été effectués pour le calcul des supports}
10: **Pour chaque** quantificateur $Q$ **Faire**
11: Calcul du degré de vérité
12: $T \leftarrow \mu_Q(Comptage(A \wedge B)/Comptage(A))$
13: {Le résumé est de la forme}
14: {*"Q des objets qui sont A sont B : T"* }
15: **Fin Pour**
16: **Fin Pour**

### 3.7.3 Propriétés des résumés flous

Les résumés flous sont caractérisés par le quantificateur employé ainsi que par les termes de résumé eux-mêmes. On utilise ici principalement deux méthodes de réduction des calculs.

*Quantificateurs*

Il est trivial de constater que si deux sous-ensembles flous représentant deux quantificateurs sont d'intersection nulle, et que le degré de vérité vaut 1 pour l'un des deux, alors il vaudra 0 pour le second. On peut élargir cette idée en considérant que dès que l'on trouve un degré de vérité important pour un quantificateur, l'estimation pour les quantificateurs disjoints (ou dont l'intersection est *presque* nulle) est rendue inutile.

Cependant, le plus coûteux en terme de calcul n'est pas le calcul du degré de vérité lui-même. En effet, celui-ci n'est qu'un calcul de degré d'appartenance d'un nombre (proportion) à une fonction représentant un quantificateur. Le calcul le plus coûteux est plutôt le calcul du cube avec les critères de restrictions. On utilise donc les propriétés applicables aux termes de résumé.

*Termes de résumé*

Même si l'utilisation des bases de données, et des cubes en particulier, pour la découverte d'exceptions est très intéressante, les utilisateurs sont principalement intéressés par les résumés décrivant le cas général, de même que la découverte des règles d'association est souvent effectuée pour une valeur de confiance forte (par exemple égale à 0.8). Or sous certaines conditions la découverte d'un résumé dont le degré de vérité est fort pour un

critère donné avec un quantificateur enclin à rendre compte des fortes proportions (plus de la moitié des objets considérés) permet de stopper la recherche pour tous les autres critères possibles.

On peut alors réduire l'ensemble des résumés candidats en ôtant tous les couples dont le premier terme est le même que celui considéré. Par exemple, si *"La plupart des objets qui sont A sont $c_1$"*, on sait que l'on n'aura pas *"La plupart des objets qui sont A sont $c_2$"*. En effet, si on sait que l'une des caractéristiques couvre plus de la moitié des exemples, on sait alors qu'aucune autre ne pourra faire de même.

**Proposition 1** *Étant donné un quantificateur $Q$ vérifiant $\forall x \in [0,1],\ \mu_Q(x) > 0 \Rightarrow x > 0.5$, et un ensemble de termes de résumé flous $\{c_i\}_{i=1}^n$ représentés par leurs fonctions d'appartenance $\mu_i$ $(1 \leq i \leq n)$ sur l'univers $U$ tel que $\forall u \in U, \sum_{i=1}^{n} \mu_i(u) = 1$.*

*On a :*
*S'il existe un résumé* "Q des objets qui sont A sont $c_i : \tau$" *tel que $\tau > 0$, alors il n'existe pas $c_j \neq c_i$ tel que* "Q des objets qui sont A sont $c_j : \tau'$ " *soit vrai avec $\tau' > 0$.*

La démonstration est faite en annexe E.

On considère un utilisateur désireux de générer les résumés du type *"La plupart des objets qui vérifient A sont B"*, où *La plupart* est un quantificateur vérifiant $\forall x, \mu_{Laplupart}(x) > 0 \Rightarrow x > 0.5$. Si la génération des résumés est effectuée après sélection des ensembles fréquents, alors la propriété ci-dessus doit être appliquée lors de la génération de ces ensembles fréquents pour s'avérer utile. En effet, son application après la phase de génération des ensembles fréquents est inutile car les opérations coûteuses de sélection et comptage auront déjà été effectuées. L'algorithme 6 montre comment améliorer l'algorithme 4 pour la génération des ensembles fréquents et des résumés de taille 1. Cet algorithme va en fait mêler les étapes de génération des ensembles fréquents et de génération des résumés pour éliminer les ensembles non pertinents avant même l'application d'opérations coûteuses de sélection et de comptage au cube.

### 3.7.4 Propriétés du modèle multidimensionnel flou

Le modèle multidimensionnel présente des avantages pour la génération des résumés flous. L'un des principaux est de fournir des hiérarchies qui autorisent l'appartenance graduelle à des niveaux de hiérarchie différents. Les avantages de ce modèle liés à la réduction du coût de la génération des résumés flous sont principalement dus à des apports technologiques tels que les performances de calcul des agrégats. Ces calculs sont d'autant améliorés que nous utilisons un modèle *MOLAP*, c'est-à-dire que les données sont physiquement stockées sous forme multidimensionnelle, contrairement aux approches *ROLAP* qui doivent accéder la base relationnelle à chaque requête sur le cube. En outre, ces calculs d'agrégats sont d'autant améliorés que certains agrégats peuvent être pré-calculés. Un choix judicieux de ces pré-calculs permet donc de trouver un équilibre optimal entre le temps de réponse aux requêtes utilisateur et la taille des données à stocker physiquement.

**Algorithme 6** Amélioration du calcul des ensembles fréquents

1: $totalcount \leftarrow$ Comptage de toutes les cellules du cube
2: $R = \emptyset$ {Ensemble des résumés}
3: {Génération des ensembles fréquents de taille 1}
4: **Pour chaque** dimension **Faire**
5: **Pour chaque** élément $e$ de la dimension **Faire**
6: Sélection sur la valeur de la dimension et comptage
7: $prop = \frac{comptage}{totalcount}$
8: **Si** $prop > min\_sup$ et $prop > min\_conf$
9: **Alors** $R \leftarrow R \bigcup \{(\emptyset, e), prop\}$
10: **sinon** Stop
11: {On sort de la boucle et on passe à la dimension suivante}
12: **Fin Pour**
13: **Fin Pour**

# Chapitre 4

# Recherche des cellules anormalement vides

Dans ce chapitre, nous soulevons une nouvelle problématique liée à la présence de cellules vides dans les bases de données multidimensionnelles. La présence de nombreuses cellules vides dans les hypercubes est considérée comme un problème. En effet, les cubes sont souvent très peu denses, ce qui rend difficile la tâche du stockage efficace de ces données.

Ce problème a été très étudié dans le cadre des bases de données statistiques. Il s'agissait alors de construire des séquences de valeurs nulles optimales (les plus longues possibles) afin d'optimiser l'application des techniques de compression lors de la linéarisation des tableaux multidimensionnels pour le stockage physique des données [SHO 82].

Cependant, la présence de cellules vides dans la base peut être vue comme révélatrice de certaines anomalies si les cellules vides sont entourées par des cellules non vides. Une cellule vide dans un cube peut avoir plusieurs causes :

- position impossible/non valide/inapplicable (l'analyste peut alors identifier de nouveaux marchés porteurs),
- valeur non disponible car égale à 0 (l'analyste peut alors identifier des défauts commerciaux pour certains produits, certains magasins *etc.*),
- donnée manquante (l'analyste peut alors qualifier les données et identifier les biais possibles introduits dans les calculs d'agrégats du fait de l'absence de certains résultats).

L'importance de la différence entre données inapplicables et données manquantes pour l'analyse de données a été soulignée dans [RAF 90], par exemple pour expliquer l'absence de résultats pour les cancers du sein chez l'homme dans une base de données statistiques médicale. Cependant, aucune solution pour retrouver automatiquement les données anormalement manquantes n'est fournie ni même envisagée.

La figure 36 illustre par exemple le cas de ventes de certains produits dans des villes américaines. Chacune des deux dimensions est hiérarchisée. On s'aperçoit qu'il n'y a aucun résultat de vente pour les tentes à San Francisco alors que les résultats des ventes pour la ville de Los Angeles pour tous les autres produits sont comparables. La mise en évidence d'une telle cellule permet de détecter une anomalie, que l'analyste peut alors exploiter pour identifier le problème. Par exemple, il peut découvrir que ce magasin ne vend pas ce produit alors qu'il serait intéressant de le proposer, en comparaison des résultats obtenus dans le magasin de la ville voisine. Il peut également découvrir que le résultat n'a pas

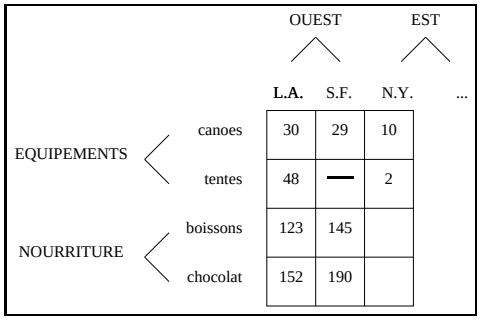


FIG. 36 – Exemple de cellule vide potentiellement intéressante

été transmis, et veiller à récupérer celui-ci, ce qui permet d'ôter un biais possible dans les analyses statistiques, introduit par cette donnée manquante. Enfin, l'analyste peut mettre en évidence un problème dans la vente de ce produit dans ce magasin, et en rechercher la cause (mise en valeur du produit etc.).

Le problème soulevé ici est donc plus général que le problème des données manquantes. Dans une autre représentation (bases relationnelles ou fichiers texte par exemple), l'analyste n'a pas de moyen pour identifier facilement des lignes anormalement manquantes, alors que le modèle multidimensionnel est parfaitement adapté à cette tâche. Cependant, l'utilisateur peut être également dangereusement tenté de considérer des données inapplicables comme des données manquantes, ce qui pourrait biaiser son analyse. Le but n'est donc pas du tout de remplacer des valeurs manquantes, mais plutôt d'attirer l'attention de l'analyste sur les cellules anormalement vides. Les méthodes proposées devront donc d'une part mettre en évidence des cellules potentiellement intéressantes et seulement elles et d'autre part s'appliquer efficacement à de très gros volumes de données. Nous proposons donc des algorithmes efficaces pour la découverte d'anomalies à partir des cellules vides.

La suite de ce chapitre est organisée comme suit : la section 4.1 présente brièvement les travaux connexes. La section 4.2 détaille l'approche par voisinage proposée ici pour la découverte des cellules vides intéressantes. La section 4.3 présente l'extension de cette approche aux bases de données multidimensionnelles floues. La section 4.4 décrit une approche par apprentissage fondée sur les arbres de décision.

## 4.1 Travaux connexes

Le problème des données manquantes est apparu dès les débuts des travaux sur les bases de données [MAI 83]. Il a également été traité dans le cadre de l'apprentissage sur des fichiers texte. Il existe de nombreuses raisons pour lesquelles une donnée peut être manquante (donnée non disponible, perdue *etc.*). La présence de ces valeurs nulles ne pose pas seulement de nombreux problèmes de représentation, mais également des problèmes au niveau des requêtes. La solution adoptée dans le modèle relationnel a été d'utiliser la valeur *nulle*. Les opérations sur les bases de données ont souvent pour réponse *vrai* ou *faux*. Dans le cas de présence de valeurs nulles, la réponse donnée est alors *inconnu*.

Des travaux ont également été proposés pour remplacer les valeurs nulles, soit par une valeur par défaut (par exemple 0), soit par une valeur obtenue par des calculs statistiques, par exemple en remplaçant par la moyenne. Le cas des valeurs manquantes est également traité dans le cadre de la fouille de données [U.M 96, QUI 91]. Cependant ces travaux se

sont attachés au remplacement des données manquantes, et ne sont donc pas applicables dans le cadre de la détection de cellules anormalement vides. En effet, ces cellules vides ne correspondent pas forcément à des données manquantes, et notre problématique est plutôt d'attirer l'attention de l'analyste qui pourra agir selon sa connaissance des données.

En ce qui concerne les modèles existants de bases de données multidimensionnelles, la gestion des cellules vides tend essentiellement à proposer un stockage efficace des données pour l'optimisation de la mémoire et des requêtes. Les cellules vides ont la valeur *nulle*. La plupart du temps, les outils de visualisation et d'analyse des données multidimensionnelles proposent le remplacement des cellules vides par une valeur typique (souvent 0).

Par exemple, `Oracle Express` fournit un grand nombre d'outils de remplacement des valeurs nulles par la valeur *zéro*. Ainsi, le paramètre `ZEROFILL` de la commande d'agrégation est utilisé pour spécifier que la valeur sur un parent après généralisation sera mise à 0 si toutes les valeurs sur les enfants sont nulles. La commande `NAFILL` est également dédiée à l'affichage d'une valeur spécifiée par l'analyste pour chaque cellule vide rencontrée. Le problème de la prise en compte ou non des valeurs nulles dans le processus d'agrégation est quant à lui considéré par l'option `NASKIP`. Dans le cas où les valeurs nulles sont prises en compte, le résultat de toute opération prenant en compte une cellule nulle sera *valeur nulle*. Ceci montre à quel point le problème est important pour l'analyse, et à quel point également celle-ci peut être biaisée par la présence de cellules anormalement vides, provenant de données manquantes mêlées à des valeurs inapplicables (ne devant pas être prises en compte).

Cependant, comme le souligne [MAR 99a], l'analyse de ces cellules vides est trop souvent oubliée. Pourtant, plusieurs explications sont possibles pour la présence de cellules vides, et la prise en compte de celles-ci conduirait à une meilleure analyse des données. En effet, comme nous l'avons déjà souligné précédemment, si on considère par exemple un cube décrivant des ventes par produit, mois, ville, la présence d'une cellule vide peut révéler soit l'indisponibilité d'un produit dans un magasin alors qu'il s'y vendrait bien, soit l'absence suspecte de ventes d'un produit dans un magasin, soit la présence d'une valeur de résultat de ventes non connue. Dans ce dernier cas, les analyses statistiques précédentes ont donc été faussées.

On note que les cubes très clairsemés ne sont pas les plus intéressants à analyser. Ils traduisent plutôt le fait que les données sont modélisées à un trop grand niveau de détail, ou qu'elles ne sont pas adaptées à la représentation multidimensionnelle [PEN 95a].

Or, à notre connaissance, aucune méthode ne traite le problème de la recherche des cellules vides potentiellement intéressantes. Nous proposons donc une méthode d'identification de ces cellules vides potentiellement intéressantes. Cette méthode est étendue aux bases de données multidimensionnelles floues.

## 4.2 Détection par étude des voisinages

On s'intéresse ici à la détection des cellules vides potentiellement intéressantes en s'appuyant fortement sur la représentation multidimensionnelle pour la définition de voisinages. On appelle *cellule vide intéressante* une cellule vide qui *dénote* à côté de cellules non vides proches (au sens des hiérarchies). Dans cette section, nous introduisons les notions de voisinage de cellule (section 4.2.1) et de degré d'intérêt d'un voisinage (section 4.2.2). Ces notions sont utilisées dans l'algorithme proposé dans la section 4.2.3. Des améliorations sont proposées par l'utilisation d'une méthode de réduction de la taille du cube à prendre

en compte (section 4.2.4), et enfin, nous apportons quelques remarques concernant les hiérarchies (section 4.2.5).

### 4.2.1 Notion de voisinage de cellule

On appelle *voisinage* d'une cellule donnée un ensemble de cellules qui sont proches au sens des hiérarchies, c'est-à-dire membres de la même catégorie le long d'une ou plusieurs dimensions. Deux formes de voisinage sont définies, qui sont détaillées ci-dessous.

#### Voisinage par bloc

On appelle *voisinage par bloc* un ensemble des cellules pour lesquelles les positions sur toutes les dimensions ont même parent. Il n'existe donc toujours qu'un seul *voisinage par bloc* pour chaque cellule. Par exemple, l'ensemble des cellules correspondant aux *équipements* pour les villes de l'*ouest* forme le voisinage de la cellule à la position *(tentes,S.F.)*.

De manière plus formelle, on considère deux niveaux de granularité dans la hiérarchie liée au cube, le niveau *parent* n'étant pas forcément parent immédiatement du niveau *fils*. On note $p(d_i)$ le parent d'un élément $d_i$ du domaine $dom_i$ de la dimension $D_i$ et on définit le voisinage d'une cellule à la position $(d_1, \ldots, d_k)$ comme étant l'ensemble des cellules $C$ aux positions $(d'_1, \ldots, d'_k)$ telles que :

$$\mathcal{V}(d_1, \ldots, d_k) = \{(d'_1, \ldots, d'_k) \mid \forall (i = 1, \ldots, k)\ p(d_i) = p(d'_i)\}$$

#### Voisinage par tranche/colonne

On appelle *voisinage par colonne* l'ensemble des cellules appartenant à des colonnes qui ont même parent dans la hiérarchie que la colonne comprenant la cellule vide considérée.

De manière plus formelle, on définit un tel voisinage pour une colonne à la position $d_i$ comme étant l'ensemble des cellules $C$ telles que :

$$\mathcal{V}(d_1, \ldots, d_i, \ldots, d_k) = \{(d'_1, \ldots, d'_i, \ldots, d'_k) \mid p(d_i) = p(d'_i)\}$$

Cette notion est étendue pour prendre en compte les voisinages constitués de plusieurs colonnes. Un voisinage d'ordre $j$ sur un ensemble de dimensions $J \subseteq \{D_i\}_{i=1,\ldots,k}$ est alors défini comme l'ensemble des cellules ayant $j$ valeurs fixées sur les dimensions de $J$ telles que pour toutes ces valeurs fixées, les parents sont les mêmes pour toutes les cellules du voisinage et pour la cellule vide. Le cas particulier où $j = k$ correspond à un voisinage par bloc.

#### Exemple

On reprend l'exemple de la figure 36. Les voisinages possibles de la cellule vide à la position $(tentes, S.F.)$ sont les suivants :

- Le voisinage par bloc correspondant à la cellule vide à la position $(tentes, S.F.)$ est l'ensemble des cellules
  $\mathcal{V}_1 = \{(canoes, L.A.), (canoes, S.F.), (tentes, L.A.)\}$
- Les deux *voisinages par tranche* de la cellule vide $(tentes, S.F.)$ sont :
  $\mathcal{V}_2 = \{(canoes, L.A.), (canoes, S.F.), (canoes, N.Y.)\}$
  $\mathcal{V}_3 = \{(canoes, L.A.), (tentes, L.A.), (boissons, L.A.), (chocolat, L.A.)\}$

### 4.2.2 Estimation de l'intérêt d'un voisinage de cellule

L'intérêt d'un voisinage est lié à la ressemblance pouvant être établie entre les cellules, qui permet de déduire qu'une cellule, trouvée vide dans le cube, aurait dû avoir une autre valeur.

On définit donc la notion de *degré d'intérêt* $\delta$. Ce degré est lié à la ressemblance des cellules entre elles. Il est calculé de manière à prendre en compte chaque lien entre les valeurs des cellules le long de toutes les colonnes des dimensions. Les domaines des dimensions sont donc combinés par ré-ordonnancements successifs des colonnes ayant même parent autour de la colonne de la cellule vide. Cette opération (*switch*) est effectuée jusqu'à trouver une colonne qui *ressemble* à la colonne dans laquelle se trouve la cellule vide. Le degré d'intérêt calculé est alors lié à une tranche, c'est-à-dire à une valeur $d_i$ de dimension $D_i$ et à une cellule vide $c$.

Par exemple, le calcul d'un des degrés d'intérêt liés au cube de la figure 36 est effectué en comparant les deux colonnes correspondant respectivement aux ventes de *Los Angeles* et de *San Francisco* pour les produits *canoes*, *tentes*, *boissons*, *chocolat*. Le calcul de ce degré peut être effectué de plusieurs manières différentes. Plusieurs méthodes possibles sont décrites et discutées ci-dessous.

#### Méthodes non floues

De nombreuses méthodes sont envisageables, mais étant donné la taille volumineuse des bases de données multidimensionnelles, une grande attention doit être portée au passage à l'échelle. On pourrait alors par exemple se contenter de calculer les valeurs absolues des différences. Cependant cette méthode n'est pas satisfaisante puisqu'elle ne prend pas en compte les différences d'amplitude entre les différentes lignes. Par exemple, il se vend beaucoup plus de nourriture que d'équipements, et il ne serait pas satisfaisant que les différences soient traitées de la même manière. Une solution est d'utiliser des ratios, et de les comparer. Si tous les ratios sont approximativement égaux, on peut en déduire que les deux colonnes sont liées et que donc la cellule vide peut être considérée comme anormalement vide.

Dans nos travaux, nous utilisons la corrélation linéaire entre colonnes, et des méthodes statistiques classiques [LEF 80].

En reprenant l'exemple précédent, on obtient le degré d'intérêt du voisinage $\mathcal{V}_3$ en comparant toutes les valeurs obtenues à *San Francisco* avec les valeurs obtenues à *Los Angeles*, pour les produits *canoes, boissons,* et *chocolat*, et en calculant la corrélation linéaire entre les colonnes *L.A.* et *S.F.* soit $0,998$ (corrélation parfaite). On en déduit donc que la cellule vide à la position *(tentes,S.F.)* est une anomalie.

#### Méthodes floues

Des relations floues sont également utilisées, par exemple *approximativement égal à*. On a alors $0 \leq \delta \leq 1$, et $\delta = \mathcal{G}_{c' \in \mathcal{V}_i(c)}(f_R(v_{c'}, v_c))$ où

- $\mathcal{V}_i(c)$ est l'ensemble des cellules du voisinage de la cellule $c$ telles que leurs positions contiennent la valeur $d_i$ pour la dimension $D_i$,
- $f_R$ est la fonction d'appartenance de la relation floue $R$,
- $\mathcal{G}$ est un opérateur d'agrégation utilisé pour fusionner les résultats des différences entre les valeurs des mesures sur les deux colonnes considérées. Une t-norme $\top$ peut par exemple être utilisée pour construire $\mathcal{G}$ en considérant

$$\mathcal{G}(x_1, ..., x_n) = \top(x_1, \top(x_2, \top(..., x_n)...)$$

Nous utilisons classiquement la t-norme *min* pour sa propriété d'idempotence.

Le problème dans ce cas est de choisir la relation floue et l'opérateur d'agrégation pour fusionner les degrés de relation entre éléments. Une cellule vide est considérée comme une anomalie s'il existe une relation entre deux colonnes, et cette relation n'est pas obligatoirement du type *forte similarité*. Par exemple, si deux colonnes sont fortement en relation par une relation du type *environ la moitié de*, alors la cellule vide est considérée comme une anomalie.

### 4.2.3 Algorithme de découverte des cellules vides potentiellement intéressantes

La découverte automatique des cellules vides potentiellement intéressantes suit le processus décrit par l'algorithme 7.

**Algorithme 7** Découverte des cellules vides intéressantes

1: Parcours des cellules vides
2: **Pour chaque** cellule vide **Faire**
3: Calcul des voisinages
4: Calcul des degrés d'intérêt
5: **Fin Pour**
6: **Pour chaque** cellule vide pour laquelle il existe un voisinage intéressant **Faire**
7: Préparation et visualisation des résultats
8: **Fin Pour**

### 4.2.4 Améliorations, prétraitements

Le but est de réduire le nombre de cellules vides à parcourir. La méthode proposée revient à isoler les *blocs* de cellules vides. Dans ce contexte, on appelle *bloc de cellules vides* un ensemble de cellules dont aucune n'a un voisinage intéressant. L'élimination de ces *blocs* de cellules vides s'effectue par une réduction des domaines actifs de dimension, en utilisant les hiérarchies pour repérer les *blocs* au sens des hiérarchies respectives.

Même si une requête *directe* est possible, la méthode de suppression des cellules vides non intéressantes suit un certain nombre d'étapes implicites, détaillées ci-dessous.

1. *Roll-Up :* on généralise le cube à un niveau de granularité plus élevé sur chacune des dimensions qui le composent. La proportion de cellules vides est alors fortement réduite par le processus d'agrégation.
2. Réduction des domaines actifs des dimensions : les domaines des dimensions du cube sont réduits aux seules valeurs dont les cellules correspondantes sont non vides. Par exemple, exprimé dans la syntaxe Oracle Express, cette requête est :

`limit` $dim_1$, ..., $dim_k$ `to cube ne NULL`

3. *Drill-Down :* on respécifie les dimensions au niveau de granularité initial. Certaines cellules vides sont encore présentes, d'autres ont disparu du fait de la réduction des domaines des dimensions. Les cellules vides qui ont disparu sont celles qui faisaient partie de blocs de cellules vides, et qui donc ne sont pas intéressantes au sens de notre définition.

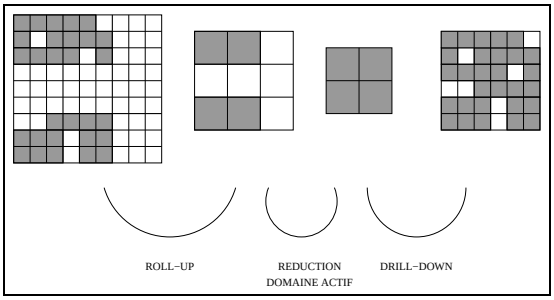


FIG. 37 – Suppression des blocs vides non intéressants

**Exemple.** La figure 37 illustre les étapes de suppression. Dans une première phase, le cube considéré est généralisé à un niveau de hiérarchie plus élevé. Le taux de cellules vides diminue alors notablement, comme l'illustre le deuxième cube de la figure. Les domaines actifs des dimensions du cube sont alors réduits aux seules valeurs pour lesquelles il existe au moins une cellule non vide, ainsi le troisième cube de la figure 37 ne contient plus que quatre cellules. Le cube est alors respécifié au niveau de granularité initial tout en conservant la réduction des domaines actifs, on obtient alors le dernier cube de la figure, qui contient beaucoup moins de cellules vides que le cube initial.

### 4.2.5 Utilisation de tous les niveaux de hiérarchies

Les hiérarchies jouent une rôle très important dans le modèle multidimensionnel, et permettent notamment la définition des concepts de voisinage (voir sections précédentes). La généralisation d'un cube permet de réduire le taux de cellules vides. Ainsi, les améliorations décrites dans la section 4.2.4 seront d'autant plus drastiques qu'elles seront effectuées à des niveaux de granularité faibles. En effet, le processus de généralisation efface la présence de cellules vides.

La possibilité de considérer tous les niveaux de hiérarchie est donc envisagée, et toutes les combinaisons niveau fils - niveau parent sont considérées, y compris les combinaisons où on n'utilise pas le parent immédiat. Cependant, on notera que l'efficacité, en terme de réduction, de l'algorithme proposé peut pâtir du mauvais choix de granularité.

De plus, plus le niveau de granularité choisi est élevé, plus le nombre de colonnes à considérer est réduit. Ainsi, ne sont prises en compte que les colonnes ayant le plus de chances d'être corrélées avec la colonne contenant la cellule vide. Il serait bien sûr possible de considérer les colonnes sur toute la dimension, cependant cette méthode serait beaucoup plus coûteuse.

En outre, selon la nature de la fonction utilisée pour le calcul du degré d'intérêt, les seuils décrits dans les sections précédentes doivent être fixés en fonction du niveau de hiérarchie et de la fonction d'agrégation utilisée. En effet, dans le cas d'une généralisation par une fonction de type *total* (somme des valeurs de toutes les cellules filles pour obtenir la valeur de la cellule parent), la définition de la différence caractéristique entre deux valeurs de cellule est dépendante du niveau de granularité.

Enfin, l'utilisation d'un niveau parent très général dans la hiérarchie induit la présence dans un voisinage de colonnes de moins en moins corrélées, ce qui n'est pas intéressant dans le cadre de ces travaux.

## 4.3 Extension aux bases de données multidimensionnelles floues

Les méthodes proposées sont étendues au cas de bases de données multidimensionnelles floues telles que décrites dans la première partie.

Une cellule vide est alors une cellule dont le degré $\mu$ est inférieur à un seuil $\epsilon$, ce seuil étant à définir. En ce qui concerne les hiérarchies, la relation de parenté est obtenue en considérant soit une relation $\prec$ de finesse entre partitions floues, soit une relation d'ordre floue. Par exemple, pour un seuil $\sigma$ donné, et une relation d'ordre floue $R$ donnée, deux cellules $C_1(d_1, \ldots, d_h, \ldots, d_k)$ et $C_2(d'_1, \ldots, d'_h, \ldots, d'_k)$ ont même parent sur une dimension $h$ si

$$\sup_{p_i=p_1,\ldots,p_n} \min(f_{R_T}(d_h, p_i), f_{R_T}(d'_h, p_i)) > \sigma$$

où $\{p_1, \ldots, p_n\}$ est l'ensemble des éléments de la dimension au niveau de granularité parent, et $f_{R_T}$ est la fermeture transitive de la relation d'ordre floue. L'algorithme est alors appliqué tel qu'il est défini précédemment.

## 4.4 Détection par apprentissage

Dans cette section, nous présentons une autre méthode non plus fondée sur les notions de voisinages, mais utilisant des méthodes de découverte d'exceptions. Toutes les méthodes d'apprentissage supervisé sont susceptibles de donner des résultats.

Nous proposons notamment de construire un arbre de décision à deux classes, l'une correspondant aux cellules vides, et l'autre aux cellules non vides. La recherche des cellules *anormalement vides* revient alors à détecter des exceptions dans les feuilles.

Deux phases sont donc nécessaires à l'application de cette méthode. La première consiste à construire l'arbre de décision en considérant deux classes possibles (vides ou non vides). La seconde consiste à étudier de manière la plus automatique possible les feuilles de l'arbre afin de déceler les branches où l'impureté est suspecte. Cette seconde phase est la plus délicate, afin que le bruit ne devienne pas l'information donnée à l'utilisateur.

## 4.5 Conclusion

Nous rappelons que notre but n'est en aucun cas de remplacer les cellules vides, mais plutôt d'attirer l'attention de l'analyste sur des cellules anormalement vides. L'analyste peut alors rechercher la cause de la présence de cette cellule vide. Plusieurs raisons peuvent alors être explorées, décrites précédemment.

L'analyste peut également utiliser cette connaissance pour retrouver la valeur à partir de la base initiale, avant le calcul du cube, si des valeurs manquantes étaient présentes non pas sur la mesure mais sur l'une des dimensions. Par exemple, le tableau ci-dessous présente une base initiale possible avant la construction du cube associé. On note que la

valeur manquante ne correspond pas à la mesure du cube, et que la mise en valeur de la cellule anormalement vide peut servir à retrouver la valeur des ventes. Par exemple, si le cube de la figure 36 a été construit à partir de la relation suivante, on peut retrouver facilement la valeur manquante et reconstituer le cube complet à partir de l'entrepôt.

| Ville | Produit | Ventes |
|---|---|---|
| L.A. | canoës | 30 |
| L.A. | tentes | 48 |
| L.A. | boissons | 123 |
| L.A. | chocolat | 152 |
| S.F. | canoës | 29 |
| S.F. | ? | 60 |
| S.F. | boissons | 145 |
| S.F. | chocolat | 190 |

# Quatrième partie

# Implantation et résultats

L'implantation des systèmes de fouille de données liés aux bases de données est une étape primordiale afin de valider les performances en termes de temps de calcul et d'optimisation de la mémoire. Les systèmes étendus au flou visent eux plutôt à optimiser la *compréhensibilité* des résultats présentés à l'utilisateur. Nous visons donc à concilier ces deux approches, afin d'obtenir les meilleures performances possibles tout en préservant la production de résultats pertinents et utiles à l'utilisateur.

Cette partie présente la mise en œuvre du modèle de représentation des connaissances et des algorithmes de fouille de données. L'introduction de la théorie des sous-ensembles flous et de requêtes flexibles dans des systèmes implantés a souvent été faite. Elle permet d'augmenter fortement la puissance des modèles et fournit aux utilisateurs des outils mieux à même de répondre à leurs attentes. Les données manipulées sont exprimées en langage naturel, et on se place ainsi dans le contexte du *computing with words* pour la représentation qualitative des données quantitatives [ZAD 96, ZAD 99]. Cette représentation, plus proche de ce que l'expert manipule habituellement, augmente la compréhensibilité des résultats d'analyse et fournit une meilleure interface pour l'exécution des requêtes entre l'utilisateur et le système.

Afin de fournir de tels outils à l'utilisateur, une sur-couche a été construite sur le système de gestion de bases de données multidimensionnelles *Oracle Express* pour la gestion de données imparfaites et leur manipulation. L'architecture de fouille de données fournit les outils d'analyse au sein de trois modules : construction d'arbres de décision, génération de résumés flous, et recherche des cellules anormalement vides.

L'ensemble de ce système est facilement utilisable par les analystes grâce aux interfaces graphiques fournissant un environnement convivial. Deux systèmes sont proposés. Le premier système, *FUB* (**FU**zzy c**UB**e), est dédié à la visualisation et la manipulation des données multidimensionnelles floues. Le second système, *FUB Miner*, permet la mise en œuvre des modules de découverte de connaissance à partir de bases de données multidimensionnelles floues. La syntaxe des langages fournis par les systèmes de gestion de bases de données pour l'implantation de fonctions internes *stored procedures* étant pauvre, la plupart des fonctionnalités de fouille de données sont externes au système de gestion de bases de données auquel elles font appel par le biais des bibliothèques (API - Application Programming Interface) autorisant l'exécution de requêtes. Ces fonctionnalités sont implantées en C++, Oracle Basic et Java.

# Chapitre 1

# Représentation et manipulation de bases de données multidimensionnelles floues

La plupart des fonctionnalités proposées dans cette thèse font l'objet d'une implantation logicielle. *FUB* est le système permettant la visualisation et la manipulation de cubes flous. *FUB* est principalement construit pour intégrer les fonctionnalités de manipulation des cubes flous en vue de son intégration dans l'architecture de *Fuzzy OLAP Mining* proposée.

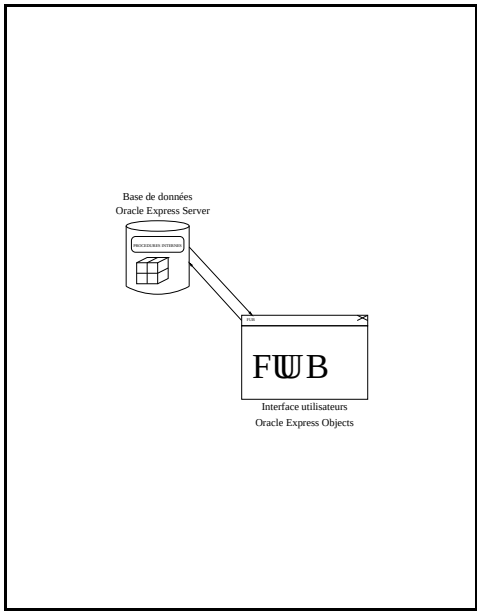


FIG. 38 – *FUB* : architecture générale

L'architecture du système *FUB* consiste en deux principaux axes (voir figure 38). Certaines des fonctionnalités sont intégrées directement dans le système de gestion de bases de données (notamment la redéfinition des opérateurs multidimensionnels), comme dans beaucoup de systèmes existants [IKE 81, KAC 95, ALB 99a, RIB 99]. L'interface utilisateur est développée à l'aide d'*Oracle Basic* (voir figure 39).

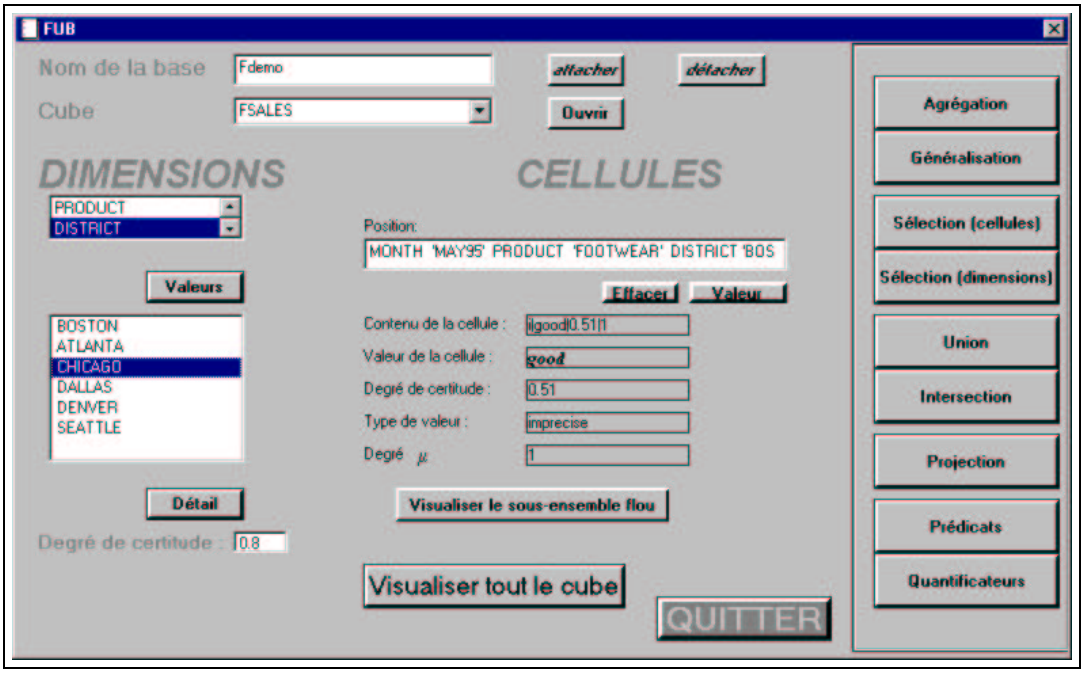


FIG. 39 – *FUB* : fenêtre principale de l'interface utilisateur

## 1.1 Représentation des données floues

Les bases de données multidimensionnelles floues sont stockées dans des bases *Oracle Express* étendues à la représentation de cubes flous et de hiérarchies floues.

Dans le cas des cubes flous, les extensions concernent la prise en compte :
- de données imprécises et/ou incertaines (éléments des cellules),
- de cellules appartenant graduellement au cube (degrés $\mu$),
- de dimensions dont les éléments appartiennent graduellement aux domaines.

Ainsi, chaque cellule du cube contient un élément constitué d'une paire (valeur, degré), ainsi que le degré d'appartenance $\mu$ de la cellule au cube. Les valeurs $d_i$ des dimensions sont associées à un degré $d(d_i)$.

Les valeurs imprécises sont représentées par un label (par exemple *bad, mid,* ou *good* dans le cas du cube flou des ventes) qui renvoie à un sous-ensemble flou. Les fonctions d'appartenance des sous-ensembles flous considérés sont linéaires par morceau. Chaque sous-ensemble flou est représenté par la suite des points décrivant les segments successifs de sa fonction d'appartenance. Par exemple, les ventes *moyennes* sont représentées par 0 ;0-30 ;0-50 ;1-110 ;1-130 ;inf ;0-. Cette suite de points correspond au sous-ensemble flou de la figure 14 de la page 48.

En outre, les hiérarchies floues sont représentées à l'aide de relations prenant en compte les deux niveaux de granularité ainsi qu'une dimension permettant la représentation du degré d'appartenance des éléments à la relation floue.

On rappelle que cette implantation permet la représentation de cubes et de hiérarchies classiques (non flous).

## 1.2 *FUB* : visualisation des données

La fenêtre principale de l'interface utilisateur permet la sélection de la base de données à explorer et du cube flou à visualiser (voir figure 39). Les cubes disponibles sont automatiquement proposés à l'utilisateur dès qu'il a choisi la base sur laquelle travailler. Une fois le cube flou choisi, ses dimensions sont affichées automatiquement. Par simple double-clic sur l'une des dimensions (ou par le bouton *Valeurs*), les valeurs de la dimension sont données.

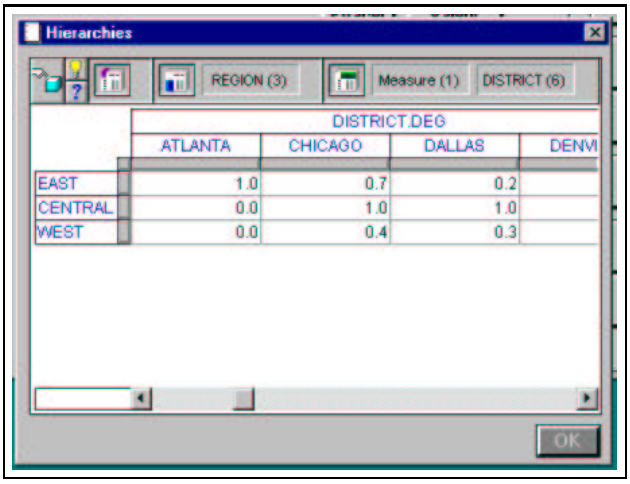


FIG. 40 – *FUB* : visualisation des hiérarchies floues

Le bouton *Détail* permet la visualisation des hiérarchies définies sur la dimension. Par exemple, la figure 40 montre la relation floue liant les villes aux différents régions.

Les degrés d'appartenance de chaque valeur de dimension au domaine sont affichés de manière automatique dès qu'une sélection est opérée par l'utilisateur.

Le cube flou peut être visualisé :
- soit cellule par cellule,
- soit de manière globale.

Dans le premier cas, l'utilisateur doit indiquer la position de la cellule à afficher en fonction des dimensions du cube (par exemple la position *MONTH 'MAY95' PRODUCT 'FOOTWEAR' DISTRICT 'BOSTON'*). Afin d'éviter une saisie fastidieuse et erronée, le système guide l'utilisateur par une fonctionnalité liée à la liste des valeurs des dimensions : dès que l'utilisateur presse la touche *'a'*, la description de la dimension et de la valeur est ajoutée à la ligne définissant la position de la cellule à visualiser.

La cellule est alors visualisée et l'utilisateur est en mesure de savoir :
- le contenu global de la cellule,
- le détail de la valeur contenue dans la cellule,
- le degré de certitude associé à cette valeur,
- le type de valeur contenue dans la cellule (précise ou imprécise),
- le degré d'appartenance $\mu$ de la cellule au cube.

Dans le second cas, à l'aide du bouton *Visualiser tout le cube*, l'utilisateur est en mesure de visualiser l'ensemble des cellules (voir figure 41). Un double clic sur l'une des cellules permet de visualiser la cellule en question par la fenêtre d'interface principale (comme précédemment).

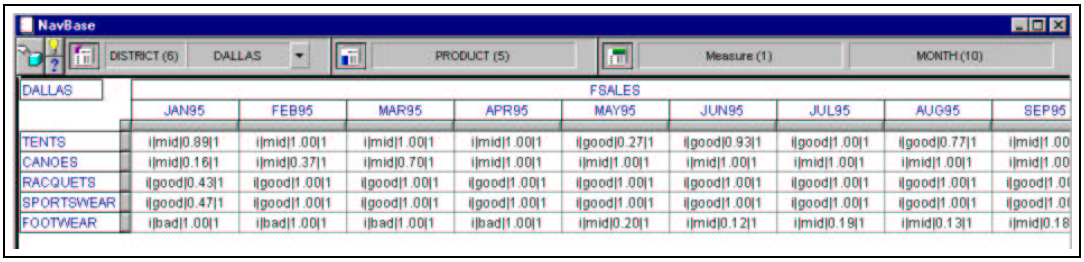


FIG. 41 – *FUB* : visualisation de la totalité du cube

Dans les deux cas, si la valeur est imprécise, le sous-ensemble flou correspondant est affiché à l'aide du bouton *visualiser le sous-ensemble flou*. Une fenêtre s'ouvre alors pour afficher la fonction d'appartenance représentant la valeur floue (voir figure 42).

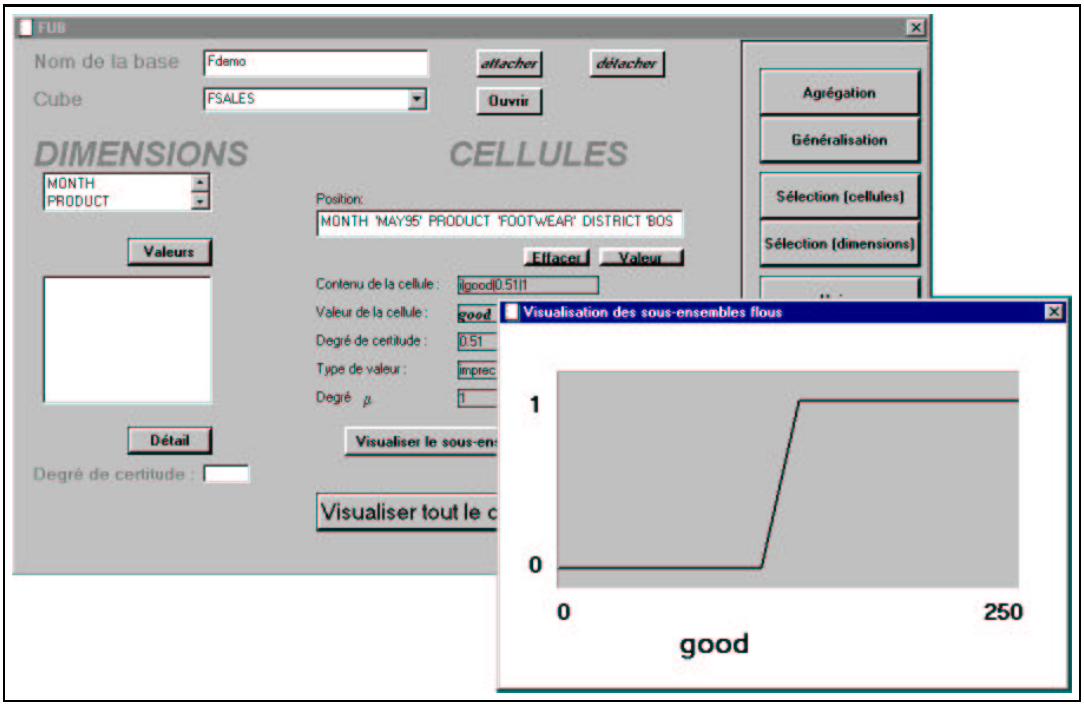


FIG. 42 – *FUB* : visualisation d'une valeur imprécise

De même que pour les valeurs imprécises contenues dans les cellules des cubes, les quantificateurs flous et les prédicats flous (utilisés ultérieurement pour la production des résumés flous) peuvent être affichés à l'aide de leurs fenêtres de contrôle respectives (voir figure 43). On note que ces sous-ensembles flous peuvent être modifiés et que de nouvelles définitions peuvent être ajoutées en utilisant les mêmes fenêtres.

## 1.3 *FUB* : manipulation des données

Les opérations implantées sont principalement dédiées à l'utilisation du modèle pour la fouille de données. Nous nous attachons donc en particulier aux opérations de :

– sélection sur les valeurs des cellules,

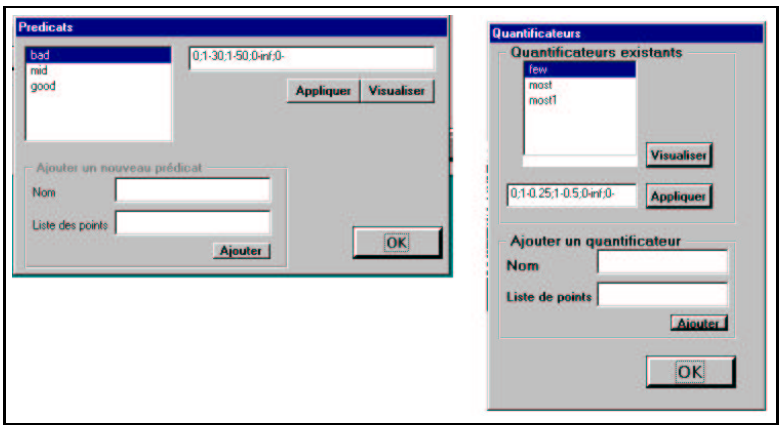


FIG. 43 – *FUB* : visualisation des prédicats et quantificateurs

– sélection sur les valeurs des dimensions,
– agrégation par comptage.

On note qu'en plus de ces fonctionnalités sont disponibles les opérations de projection et de généralisation. Les sections ci-dessous décrivent ces différentes fonctionnalités.

Dans tous les cas, l'utilisateur saisit les paramètres (si nécessaire) de l'opérateur ainsi que le nom du cube résultat.

### 1.3.1 Sélection sur les valeurs des cellules

L'opération de sélection sur les valeurs des cellules consiste à modifier les degrés d'appartenance $\mu$ des cellules au cube. Cette opération est appliquée sur un cube et requiert la définition d'un critère de sélection. L'utilisateur choisit donc l'un des critères de sélection disponibles. Ceux-ci peuvent être visualisés et édités grâce à une autre fenêtre, comme décrit précédemment.

Le programme se charge alors de parcourir l'ensemble du cube et, pour chaque cellule, de calculer le nouveau degré $\mu$ en fonction :

– de l'ancien degré $\mu$ de la cellule,
– du degré avec lequel la valeur de la cellule satisfait le critère,
– du degré de confiance en la valeur de la cellule.

Le calcul du degré de satisfiabilité de la valeur par rapport au critère est effectué en utilisant la formule (2.1) (page 58). Selon le type de valeur de la cellule (précise ou imprécise), le degré de satisfiabilité est calculé soit par la simple recherche du degré d'appartenance de la valeur de la cellule au critère, soit en calculant d'abord le sous-ensemble intersection du critère et de la valeur puis en calculant le ratio des mesures des ensembles flous concernant d'une part cette intersection valeur critère et d'autre part la valeur seule. L'intersection des sous-ensembles flous est calculée en utilisant la t-norme min. Les points des fonctions d'appartenance des deux sous-ensembles flous sont parcourus simultanément afin de repérer les différents points de coupure.

Le degré de satisfiabilité obtenu est ensuite intégré dans le calcul du nouveau degré $\mu$ de la cellule en le comparant à l'ancien degré $\mu$ et au degré de confiance en la valeur de la cellule. Le minimum de ces trois degrés est retenu comme valeur résultat.

### 1.3.2 Sélection sur les valeurs des dimensions

Le nouveau degré $d(d_i)$ de chaque tranche du cube est calculé en fonction de la formule (2.3) (page 62). Pour les calculs, les mêmes procédés que pour la sélection sur les valeurs de cellules sont utilisés.

### 1.3.3 Agrégation par comptage

L'agrégation par comptage produit comme résultat un nombre, entier ou réel selon le mode de comptage (voir équations (2.5) et (2.6) page 67). Comme exposé dans cette thèse, deux types de comptage sont possibles :

- $\Sigma$-comptage (somme des degrés $\mu$),
- comptage seuillé (comptage des cellules dont le degré $\mu$ dépasse un seuil fixé par l'utilisateur).

Un troisième type de comptage est également proposé. Construit à partir des deux précédents, il consiste à sommer les degrés $\mu$ des cellules qui sont supérieurs à un seuil fixé ($\Sigma$-comptage seuillé).

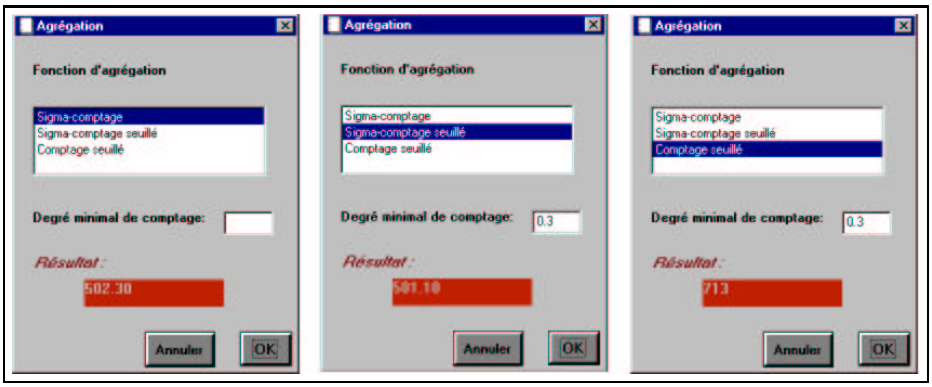


FIG. 44 – *FUB* : résultats d'agrégation par comptage

La figure 44 illustre les différents résultats possibles pour un même cube.

Le calcul du résultat est effectué par un parcours de l'ensemble des cellules du cube. Pour chaque cellule, le comptage est mis à jour en fonction du degré $\mu$ et des degrés associés à chacune des tranches dont la cellule fait partie.

### 1.3.4 Projection

La projection d'un cube sur un sous-ensemble de ses dimensions n'est pas modifiée par l'introduction du flou. L'utilisateur sélectionne les dimensions sur lesquelles le cube doit être projeté. Le système vérifie que les dimensions non spécifiées sont bien réduites à un singleton (dans le cas contraire, un message d'erreur est affiché). Il procède ensuite au remplissage du nouveau cube (dont le nom doit être spécifié par l'utilisateur) après avoir créé ce cube.

La projection du cube $C$ défini sur les dimensions $D_1, \ldots, D_k$ sur les dimensions $D_m, \ldots, D_n$ est donc effectuée en utilisant les commandes Oracle Express de la forme :

define NouvCube variable $<D_m, \ldots, D_n>$ ;
NouvCube = $C$

# Chapitre 2

# *FUB Miner* : généralités

Le système *FUB Miner* met en œuvre les fonctionnalités de fouille de données présentées dans la troisième partie de cette thèse.

## 2.1 Architecture

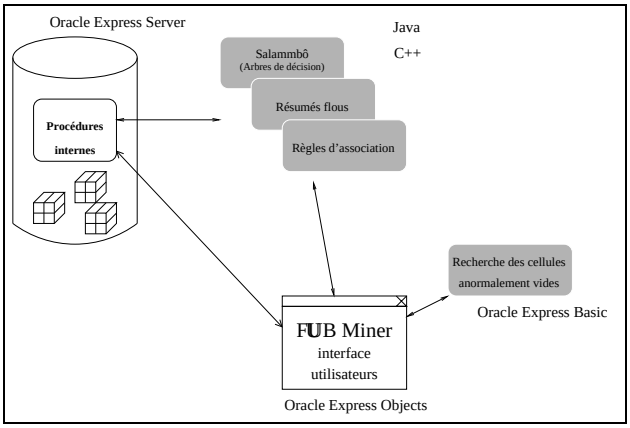


FIG. 45 – *FUB Miner* : architecture générale

L'architecture de ce système repose sur la collaboration de différents modules (voir figure 45) :

- l'interface utilisateur,
- les programmes de production des résumés flous, règles d'association et arbres de décision,
- les procédures internes aux bases de données (*Stored Procedures* Oracle) étendant les opérateurs multidimensionnels classiques au cas flou.

La communication entre ces différents modules s'effectue par différents moyens, notamment en utilisant

- *SNAPI* (**S**tructured **N**-dimentional **A**pplication **P**rograming **I**nterface) pour les communications entre les programmes C++ et Java et la base de données,
- des sockets pour les communications entre l'interface utilisateur et Salammbô,
- le passage de paramètres au cours des appels des programmes de génération des règles d'association et de résumés flous,

– le presse-papier pour la collecte des résultats de ces appels de programme.

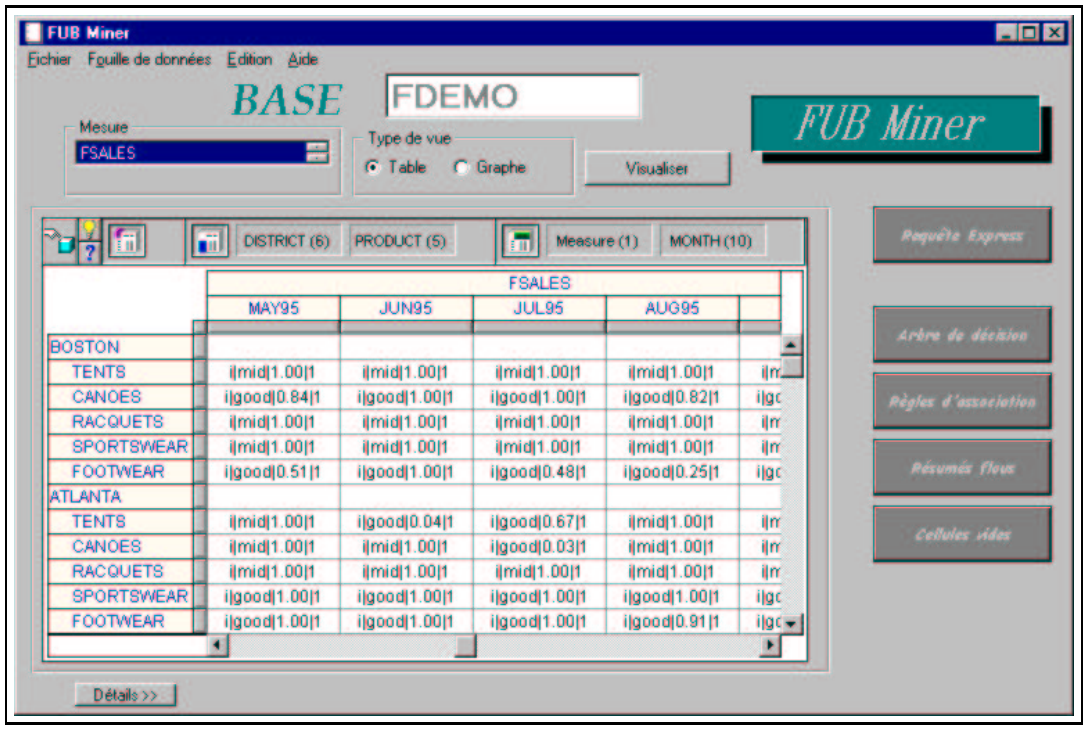


FIG. 46 – *FUB Miner* : fenêtre principale de l'interface utilisateur

## 2.2 Fonctionnalités et interface

L'interface utilisateur de *FUB Miner* repose sur les fonctionnalités suivantes (voir figure 46) :

– choix de la base de données à considérer (ainsi que la gestion de la connexion et déconnexion au serveur de bases de données multidimensionnelles Oracle Express Server),
– visualisation du cube,
– sélection des valeurs de dimension à prendre en compte lors de la fouille de données,
– sélection et paramétrage des modules de fouille de données.

Dès l'ouverture d'une base de données, la liste de tous les cubes disponibles est mise à jour. L'utilisateur choisit alors le cube sur lequel travailler par simple sélection (en pressant la touche de validation, en double-cliquant, ou en utilisant le bouton *Visualiser*). La navigation dans la base et la sélection des valeurs de dimension s'effectuent aisément à partir de la visualisation du cube ainsi obtenue.

Quatre modules de fouille de données sont proposés à l'utilisateur (quatre boutons et quatre commandes dans le menu *Fouille de données* sont disponibles). Les trois principaux (ayant fait l'objet d'une attention particulière dans cette thèse) sont présentés dans les chapitres suivants.

# Chapitre 3

# Construction d'arbres de décision flous

Les propositions exposées dans le chapitre III.2, concernant la construction d'arbres de décision flous, sont implantées au sein de *FUB Miner* à l'aide de Salammbô, dont le code est modifié pour prendre en compte l'accès aux bases de données multidimensionnelles plutôt que le chargement en mémoire centrale de la base d'apprentissage. Dans ce chapitre, nous détaillons la mise en œuvre de ce module ainsi que les résultats obtenus.

## 3.1 Mise en œuvre

La construction des arbres de décision est réalisée à l'aide de plusieurs programmes interagissant, avec principalement Salammbô [MAR 98b]. Le premier est l'interface utilisateur à partir de laquelle sont réalisées les sélections de dimensions et de valeurs sur les dimensions. Cette interface interagit par l'appel d'un programme java avec le programme C++ d'exécution des requêtes sur la base. Le programme java est utilisé notamment pour la synchronisation des connexions entre le programme C++ et Salammbô (les connexions sont réalisées à l'aide de *sockets*). Il est également utilisé pour transmettre à Salammbô les méta-données sur le cube à analyser (*e.g.* nom des dimensions, valeurs des dimensions, restriction sur les valeurs à prendre en compte). Après cette initialisation, le programme C++ est ensuite guidé par Salammbô qui transmet des requêtes à exécuter sur des sous-cubes de la base multidimensionnelle. L'interface C++ se charge alors de récupérer les résultats (de comptage) et de les transmettre à Salammbô pour chaque dimension et pour chaque valeur de classe. Salammbô est un serveur exécuté de manière permanente sur une station UNIX. Un processus fils est lancé pour chaque arbre à construire, ce qui rend le système autonome et prêt à fonctionner en mode multi-utilisateurs.

L'interface utilisateur générale est illustrée par la figure 47. Pour l'appel de Salammbô, deux modes d'utilisation sont proposés.

L'utilisateur non expert n'a besoin de spécifier que les dimensions à prendre en compte et leur type. Plusieurs types de dimension (l'utilisateur choisit l'un de ces types grâce aux boutons radio) sont possibles selon que celle-ci :

- ne doit pas être prise en compte,
- est de type entier (une partition peut être construite),
- est de type réel (une partition peut être construite),
- est de symbolique (la dimension ne doit pas être partitionnée).

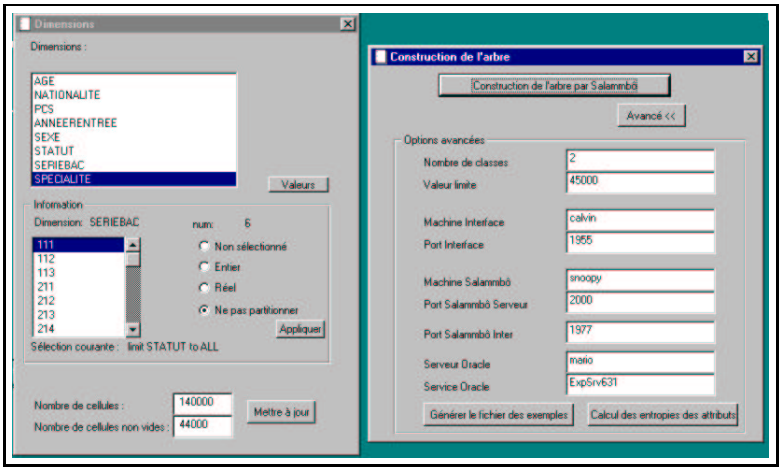


FIG. 47 – *FWB Miner* : construction d'arbres de décision flous

L'utilisateur expert est en mesure de spécifier, outre les renseignements sur les dimensions décrits ci-dessus, les noms de machines (ou adresses IP) ainsi que les numéros de ports de communication à utiliser. Ces options ne sont visibles que par le choix explicite de l'utilisateur en utilisant le bouton *Détail* de la fenêtre de droite (figure 47).

Dans les deux cas, la construction de l'arbre est amorcée par simple clic sur le bouton *Construction de l'arbre par Salammbô*. Les arbres résultats sont automatiquement présentés à l'utilisateur au format HTML. Différents types de présentations, plus ou moins détaillées, sont possibles, comme l'illustrent les figures 48 et 49. Par exemple, le détail des sous-ensembles flous construits par Salammbô est visible sur la figure 49 et pas sur la figure 48.

La construction automatique des partitions floues sur les univers numériques est effectuée par Salammbô. Le principe de cette construction repose sur l'utilisation des opérateurs de la théorie de la morphologie mathématique représentés à l'aide de la théorie des langages formels. La distribution des valeurs de chaque exemple de la base d'apprentissage sur la dimension à partitionner est étudiée en fonction de la classe associée. Les fonctions d'appartenance de référence sont alors extraites grâce au regroupement des valeurs ayant la même classe et à l'application des opérateurs d'érosion et de dilatation de la morphologie mathématique pour le filtrage (lissage notamment) des sous-ensembles flous (voir [MAR 98b] pour le détail de ces méthodes). La figure 50 résume ce processus.

## 3.2 Résultats

Les cubes utilisés sont décrits dans l'annexe G. Sans restriction sur les valeurs des dimensions, le plus gros cube considéré compte plus de 17 millions de cellules.

La dimension continue des âges est partitionnée automatiquement par Salammbô. Par exemple, des règles floues obtenues avec cette base concernent la proportion de candidats reçus au bac avec la mention AB, B ou TB :

**R1 :** Chez les jeunes*, sans spécialité, la proportion de candidats reçus avec mention (AB, B ou TB) est supérieure à 25%.

**R2 :** Chez les lycéens d'âge moyen*, sans spécialité, la proportion de candidats reçus avec mention (AB, B ou TB) est inférieure à 25%.

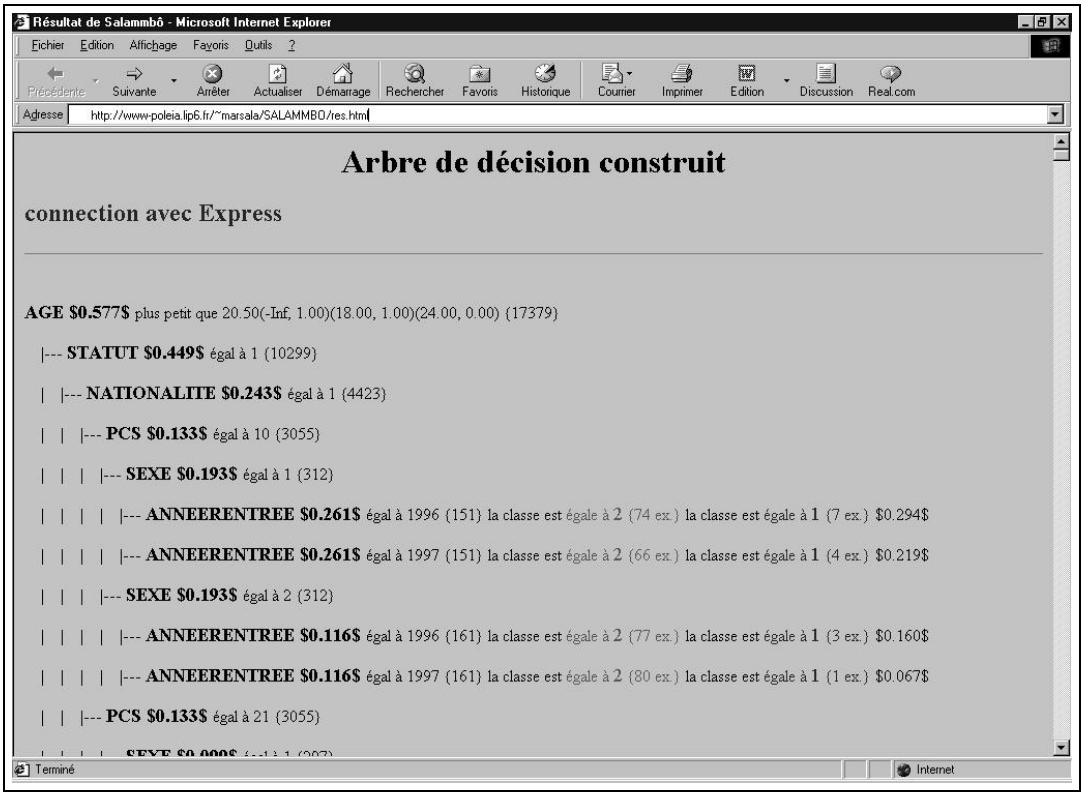


FIG. 48 – Visualisation de l'arbre construit, niveau détaillé

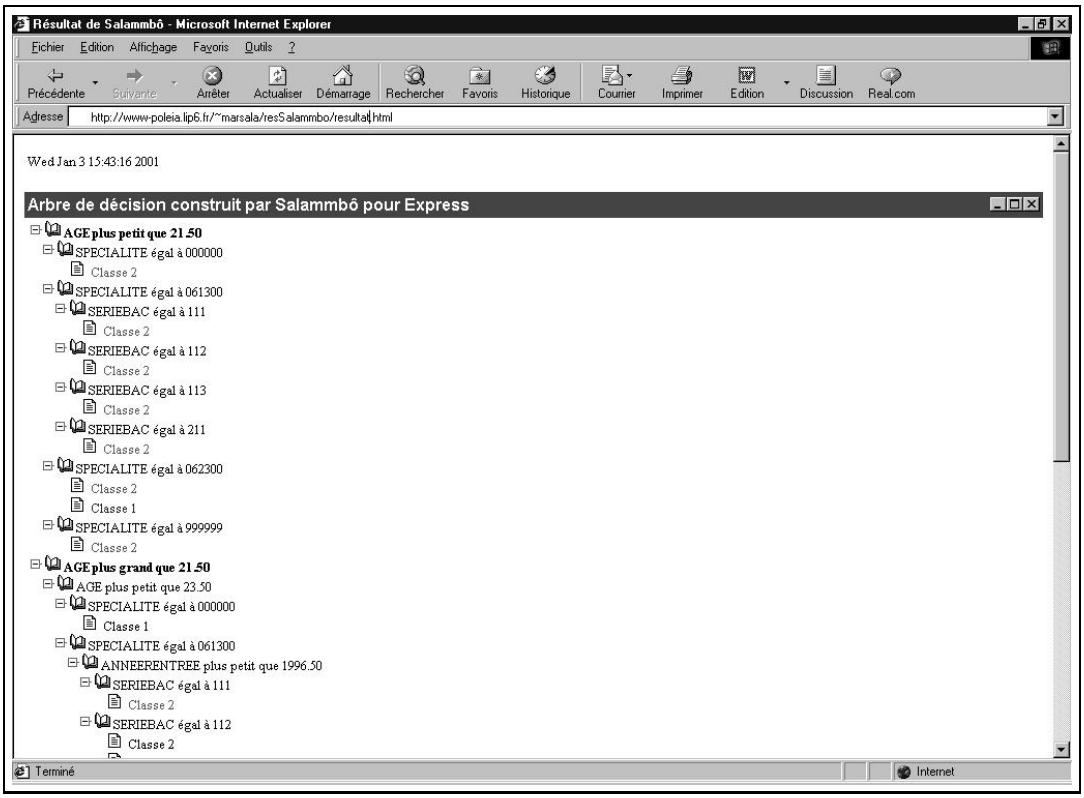


FIG. 49 – Visualisation de l'arbre construit, niveau non détaillé

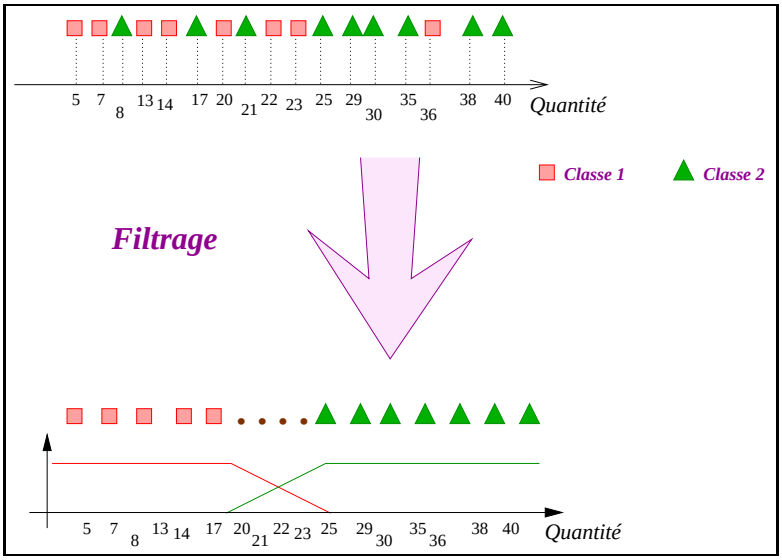


FIG. 50 – Salammbô : construction des sous-ensembles flous

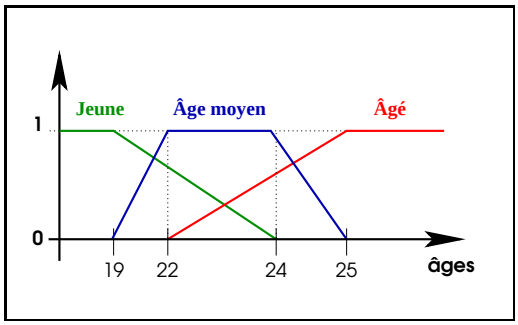


FIG. 51 – Fonctions d'appartenance construites

| Cube | Nbre de dimensions | Nbre de cellules | Nbre de cellules non vides | Densité | Tps de construction |
|---|---|---|---|---|---|
| **BASE VENTES** | | | | | |
| demo | 3 | 8 | 8 | 1 | 201 ms |
| demo | 3 | 60 | 60 | 1 | 5 s |
| demo | 3 | 120 | 120 | 1 | 10 s |
| demo | 3 | 300 | 300 | 1 | 1 min |
| demo | 3 | 540 | 540 | 1 | 1 min 40 s |
| demo | 3 | 720 | 720 | 1 | 2 min 17 s |
| **BASE BAC** | | | | | |
| CDeg | 6 | 32 | 32 | 1 | 4 s |
| CDeg | 6 | 192 | 111 | 0,58 | 17 s |
| CDeg | 6 | 1440 | 365 | 0,25 | 41 s |
| CDeg | 6 | 3136 | 446 | 0,14 | 48 s |
| CDeg | 6 | 8400 | 886 | 0,1 | 2 min |
| CDeg | 6 | 10080 | 1024 | 0,1 | 2 min 27 s |
| PAdm | 8 | 17640 | 3721 | 0,21 | 10 min |
| PAdm | 8 | 23040 | 4371 | 0,19 | 12 min |
| PAdm | 8 | 25200 | 5349 | 0,21 | 14 min |
| PAdm | 8 | 12000 | 5832 | 0,1 | 12 min |
| PAdm | 8 | 8000 | 6342 | 0,79 | 17 min |
| PAdm | 8 | 11520 | 7477 | 0,65 | 14 min |
| PAdm | 6 | 1232000 | 17379 | 0,01 | 62 min |
| PAdm | 8 | 836352 | 18111 | 0,02 | 14 min |
| PAdm | 5 | 17.297.280 | 64608 | 0,004 | 3 h |

TAB. 4 – Construction d'arbres de décision flous : résultats d'exécution

**R3 :** Chez les jeunes*, avec la spécialité "math", passant le bac série "S", la proportion de candidats reçus avec mention (AB, B ou TB) est supérieure à 25%.

**R4 :** Chez les lycéens âgés*, passant le bac série "S", la proportion de candidats reçus avec mention (AB, B ou TB) est inférieure à 25%.

* Les fonctions d'appartenance correspondantes, construites par Salammbô, sont données dans la figure 51.

Les temps de calcul nécessaires à la construction des arbres de décision dépendent de la taille du cube (en nombre de cellules), de sa densité et du nombre de dimensions à partitionner. On note que les codes source utilisés sont non optimisés, l'objectif du système étant en premier lieu de vérifier la faisabilité de la solution proposée. Globalement, les temps de construction oscillent entre quelques millisecondes et plusieurs heures, comme le montre la tableau 4. La machine utilisée pour les tests est un Pentium 700 MHz avec 256 Mo de mémoire vive.

# Chapitre 4

# Génération de résumés flous

La construction des résumés flous est assurée par un programme C++ interagissant avec la base Oracle. Ce programme est appelé à partir de l'interface graphique utilisateur Oracle Express Objects (voir figure 54) et renvoie le résultat en utilisant le presse-papier windows, ce qui évite les entrées/sorties coûteuses liées à l'utilisation de fichiers.

La génération et le stockage des fréquents avec leurs supports sont issus des algorithmes classiques des systèmes habituels de génération de règles d'association. Ces méthodes ont été totalement ré-implantées afin de prendre en compte le caractère multidimensionnel des données. Des améliorations peuvent être apportées en vue de meilleures performances en temps de calcul, un simple algorithme de type APriori étant utilisé [BAS 02].

Dans ce chapitre, nous présentons les fonctionnalités de *FUB Miner* liées à la production de résumés flous et les résultats obtenus.

## 4.1 Prise en compte de la mesure du cube

Dans la production des résumés flous (de même que pour la production de règles d'association) à partir de bases de données multidimensionnelles, la mesure du cube peut être prise en compte de deux manières différentes.

Ainsi, une mesure additive permet de considérer les valeurs des cellules comme expression du support. Par exemple, dans la base des ventes, chaque valeur de cellule représente le nombre total d'unités vendues pour le produit, la ville et le mois spécifiés par les valeurs des tranches. Dans la base BAC, le cube *PAdm* contient le nombre total de candidats admis pour chaque combinaison possible de valeurs des dimensions.

Dans ce cadre, le support d'une combinaison de critères ne dépendant pas de l'une des dimensions est aisément obtenu en sommant les valeurs des cellules de long de cette dimension.

Une mesure non additive doit être traitée avec attention. Dans notre système, ce type de mesure est traité comme une dimension supplémentaire à considérer dans les termes du résumé. Les valeurs sur cette dimension sont obtenues en considérant une partition floue de l'univers. Les termes de résumé possibles sur la mesure du cube sont donc exprimés en langage naturel. Leur nombre est fini et varie selon la finesse de la partition considérée.

*FUB Miner* travaille à partir d'une partition floue connue, cependant des techniques de construction automatique sont possibles, notamment à l'aide de Salammbô, déjà utilisé pour la construction d'arbres de décision flous.

Dans le cas de cubes flous comme de cubes classiques, une mesure additive peut néanmoins être considérée comme une dimension supplémentaire, l'utilisateur devant dans ce cas cocher l'option *Partition floue sur la mesure* de la fenêtre interface.

Dans l'état actuel de l'implantation de *FUB Miner*, l'ensemble des cubes exemples sont construits à partir de fonctions additives. L'utilisateur a le choix de traiter les cubes classiques comme il le désire. Cependant, les cubes flous sont traités uniquement comme des cubes dont la mesure est une dimension apparaissant dans les résumés. L'implantation du traitement des cubes flous de manière additive est en cours, la somme de sous-ensembles flous étant facilement calculable, notamment dans le cas de fonctions d'appartenance trapézoïdales en utilisant les résultats de l'arithmétique floue [KAU 91, HAN 99]. Les termes de la partition floue sont connus, ce sont les prédicats cités dans le chapitre IV.1.

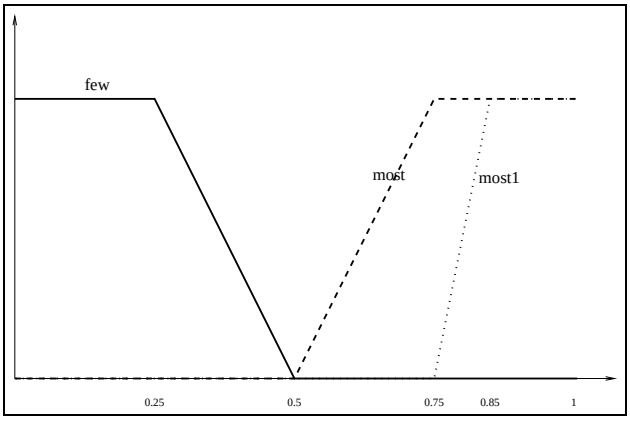


FIG. 52 – Quantificateurs utilisés

On considère les quantificateurs décrits par la figure 52. *Most* et *Most(1)* sont deux quantificateurs différents représentant la notion de *La plupart*, plus ou moins enclins à considérer de faibles proportions.

La figure 53 présente le résultat de la recherche des résumés flous sur le cube de la base des ventes décrivant le montant de l'investissement publicitaire (cube $ADVERTISING$). Par exemple, on obtient le résumé suivant :

*Peu de publicité concernant les* canoës *est effectuée à* Boston

Les sections suivantes présentent les résumés multi-niveaux et le raffinement de résumés, on note que la production de ces résumés est indépendante du choix de l'utilisateur sur la prise en compte de la mesure. L'utilisateur peut donc choisir de produire des résumés multi-niveaux et de les raffiner en considérant la mesure comme une dimension partitionnée ou non.

## 4.2 Résumés multi-niveaux

On appelle résumés multi-niveaux des résumés produits à différents niveaux de granularité, par exemple associant le niveau le plus fin des *mois* sur la dimension temporelle et le niveau élevé des *régions* sur la dimension spatiale.

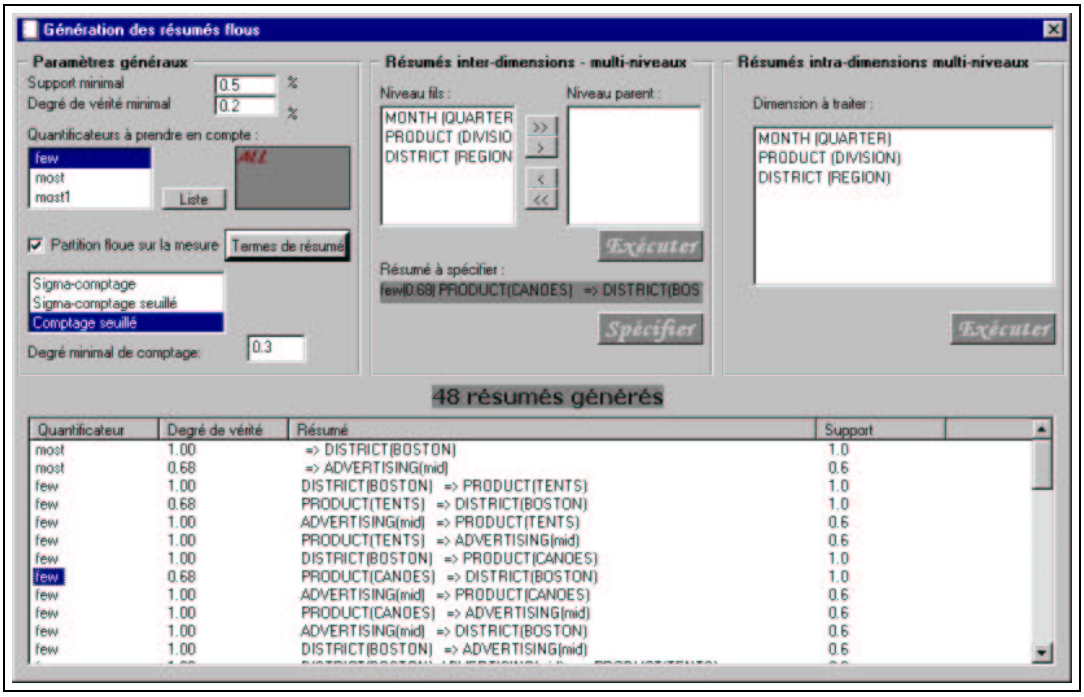


FIG. 53 – Résumés avec partition floue de la mesure

### 4.2.1 Mise en œuvre

L'utilisateur a la possibilité de sélectionner pour chacune des dimensions hiérarchisées du cube le niveau auquel il souhaite générer les résumés. Il sélectionne les valeurs des dimensions sur lesquelles il souhaite générer les résumés sur la fenêtre principale. Le système appelle alors l'exécutable C++. Les paramètres de commande sont les suivants :

- nom de la base de données,
- nom du cube,
- adresse IP du serveur Oracle Express,
- nom du service Oracle Express Server,
- commande de limitation des valeurs de dimension,
- seuil minimal de support,
- seuil minimal de degré de vérité,
- quantificateurs flous à prendre en compte,
- nom du niveau de hiérarchie à considérer pour chaque dimension.

Les résumés produits sont affichés dans la liste sur la fenêtre. Pour chaque résumé, les informations suivantes sont fournies :

- le quantificateur flou représentant le mieux le résumé flou (celui pour lequel le résumé a le plus fort degré de vérité),
- le degré de vérité,
- le résumé lui-même,
- le support du résumé.

Ces informations sont affichées chacune dans une colonne, l'utilisateur ayant la possibilité de trier les résumés selon chacune des colonnes selon son choix. Le tri est ordonné de manière ascendante ou descendante alternativement à chaque sélection successive de la

même colonne. Cet utilitaire est particulièrement utile à l'utilisateur pour sélectionner le résumé qu'il veut détailler, comme nous le décrivons dans la section 4.3.

### 4.2.2 Résultats

Le choix d'un niveau de granularité sur les dimensions permet de réduire le nombre de résumés produits, comme le montre le tableau suivant à support fixé égal à 0.12. Ces rèsultats sont dus au fait que les seuils de support sont plus facilement atteints à des niveaux de granularité élevés.

| Nombre de dimensions agrégées | Nombre de résumés générés |
|---|---|
| 3 | 45 |
| 2 | 41 |
| 1 | 16 |
| 0 | 12 |

Le choix du support permet également de réduire le nombre de résumés produits :

| Seuil de support | Nombre de résumés générés |
|---|---|
| 0.1 | 1250 |
| 0.12 | 994 |
| 0.13 | 194 |
| 0.2 | 38 |
| 0.25 | 31 |
| 0.3 | 12 |
| 0.35 | 7 |
| 0.4 | 1 |
| 0.45 et au delà | 0 |

## 4.3 Raffinement de résumés

Le raffinement de résumés permet de détailler les résultats de la production de résumés à un niveau élevé sur un niveau plus fin.

Ce processus correspond aux spécificités de l'OLAP, une aide à l'analyse de la base par la navigation étant ainsi fournie pour parcourir les différents niveaux de hiérarchie.

### 4.3.1 Mise en œuvre

Aprés production des résumés à un niveau agrégé, l'utilisateur est en mesure de sélectionner un résumé et la ou les dimensions à raffiner (voir figure 54). Le processus de génération des résumés est alors relancé sur le sous-ensemble de données correspondant au résumé sélectionné. Les dimensions que l'utilisateur veut raffiner sont alors prises en compte au niveau d'agrégation inférieur.

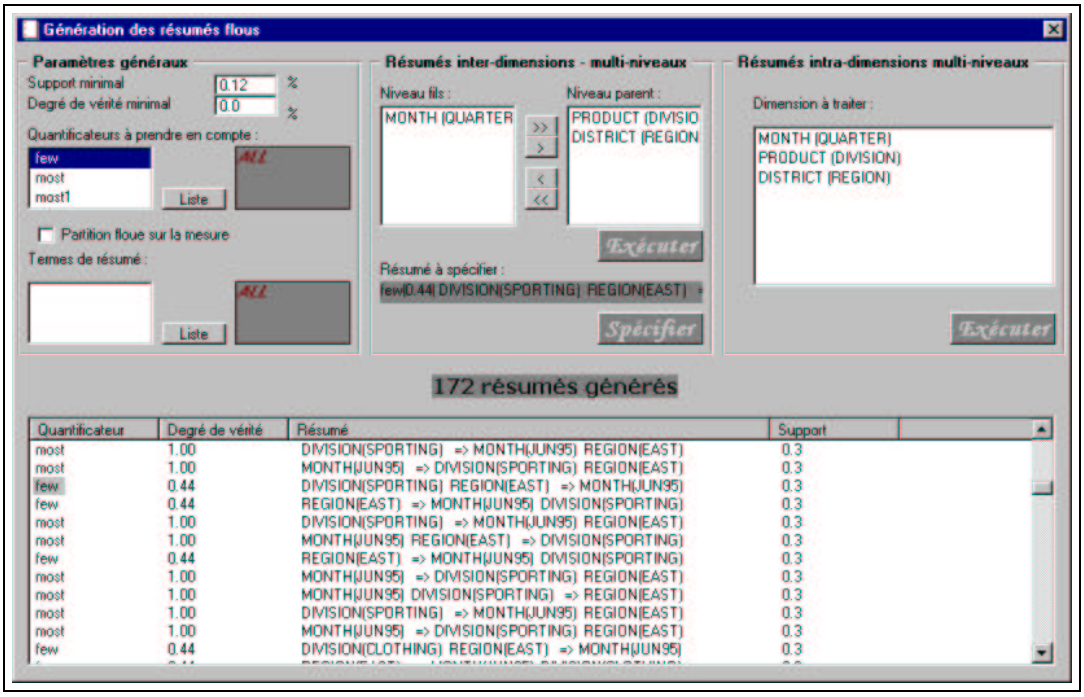


FIG. 54 – Résumés multi-niveaux et raffinement de dimension

### 4.3.2 Résultats

Par exemple, l'utilisateur sélectionne le résumé

*Peu de ventes de la région* **Ouest** *concernent la catégorie des produits de* **camping**

et souhaite le raffiner sur la dimension *PRODUIT*.

Le sous-cube correspondant aux données concernant l'Ouest et les articles de camping est alors calculé, et les résumés obtenus sont du type :

**Peu de** *ventes au* **deuxième trimestre 1995** *concernent des* **tentes**
**La plupart** *des ventes de* **tentes** *concernent le* **deuxième trimestre 1995**
**Peu de** *ventes de* **canoës** *dans la région* **Ouest** *concernent le* troisième trimestre 1995

## 4.4 Résumés intra-dimensions

Les résumés intra-dimensions permettent d'exprimer en langage naturel la répartition des données au sein d'une même dimension en fonction de la hiérarchie. Par exemple, *la plupart des ventes à l'*Est *se produisent à* Boston.

### 4.4.1 Mise en œuvre

La production des résumés intra-dimensions est effectuée après sélection d'une dimension possédant une hiérarchie. La sélection de l'utilisateur n'est pas valide si aucune dimension n'est sélectionnée, ou si aucune hiérarchie n'est définie sur la dimension choisie. Dans ce cas, un message d'erreur est affiché invitant l'utilisateur à renouveler son choix.

Quand une dimension possédant une hiérarchie est sélectionnée, le système parcourt le cube en procédant par sélections successives des sous-cubes correspondant à chacune des

valeurs de la dimension au niveau de granularité le plus élevé (par exemple la valeur *Est*). Pour chacun de ces sous-cubes, le système calcule pour chaque valeur fille du niveau le plus faible (par exemple *Boston*) la proportion d'éléments du cube et l'exprime à l'aide du quantificateur flou ayant le plus fort degré de vérité. L'ensemble des réponses est ensuite affiché dans la liste prévue à cet effet.

### 4.4.2 Résultats

Les résumés intra-dimensions permettent d'évaluer l'influence de chacune des modalités d'une dimension sur la mesure. Par exemple, on étudie sur le cube des ventes l'influence du mois au sein de chacun des trimestres.

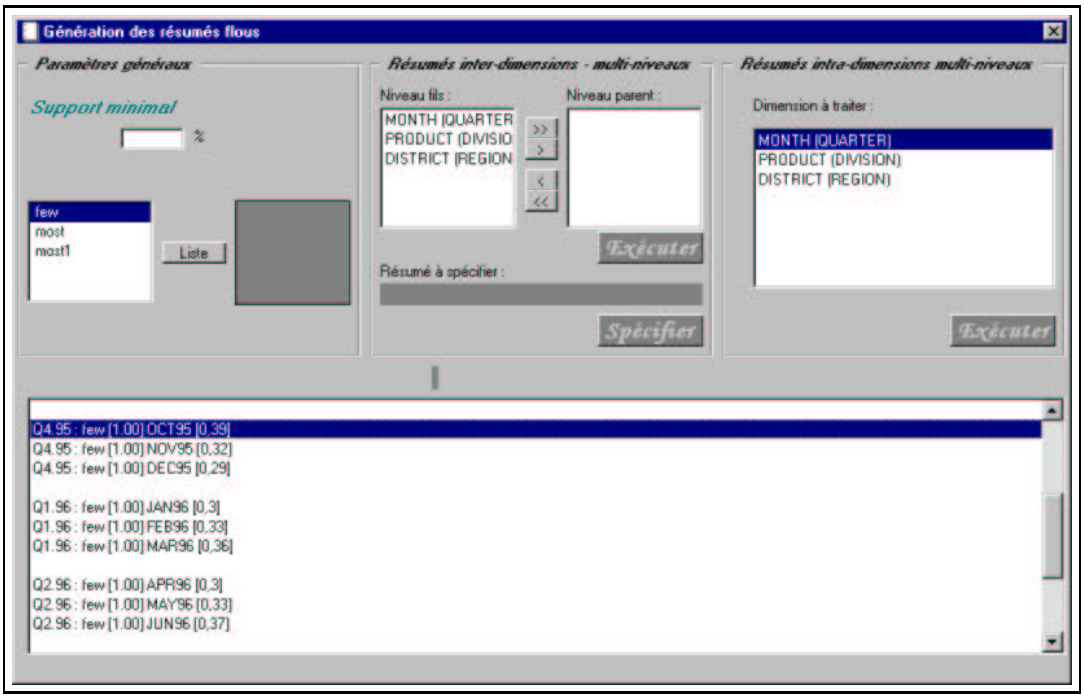


FIG. 55 – Résumés intra-dimensions : base des ventes

La figure 55 donne les résultats pour la base des ventes. Cependant, cette base étant une base jouet, les résultats sont peu intéressants car les résultats de ventes ont été répartis de manière uniforme le long des modalités des dimensions.

Sur la base BAC, des résultats plus intéressants sont obtenus, présentés sur la figure 56. Ces résultats sont obtenus à partir du cube décrivant le nombre d'admis (toutes mentions confondues). On peut par exemple en déduire que chez les plus jeunes candidats (ayant entre 13 et 16 ans), la plupart des candidats admis ont 16 ans.

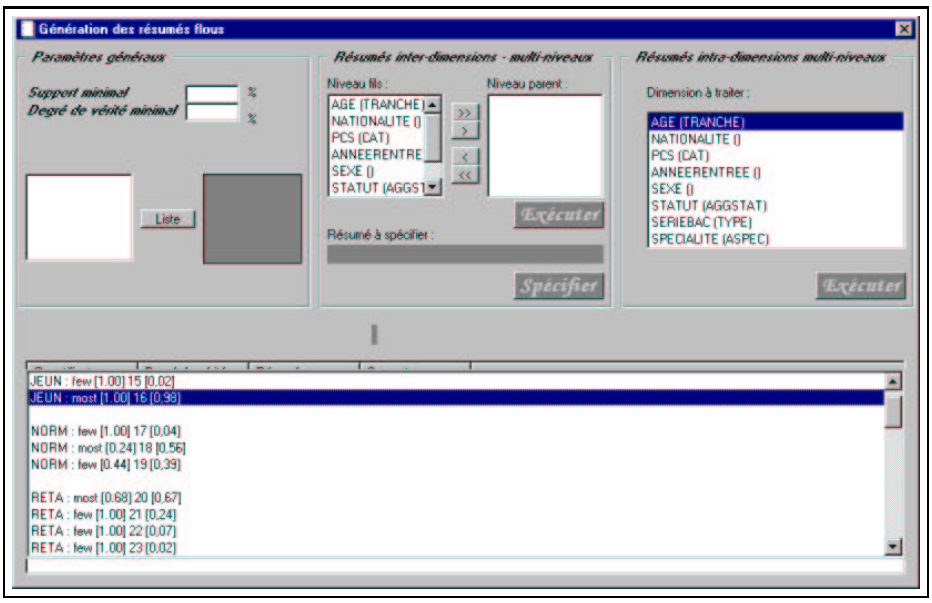


FIG. 56 – Résumés intra-dimensions : base BAC

# Chapitre 5

# Détection de cellules anormalement vides

Nous rappelons qu'aucun autre système n'existe pour la détection de cellules anormalement vides. Il ne s'agit donc pas là d'étudier de quelconques performances en termes de temps de calcul mais bien de souligner la pertinence des résultats découverts par nos algorithmes.

Pour l'étude des cellules vides, l'utilisateur a le choix entre deux méthodes, l'une reposant sur la construction de l'arbre de décision discriminant les cellules vides des cellules non vides, la seconde reposant sur le calcul des voisinages (voir figure 57).

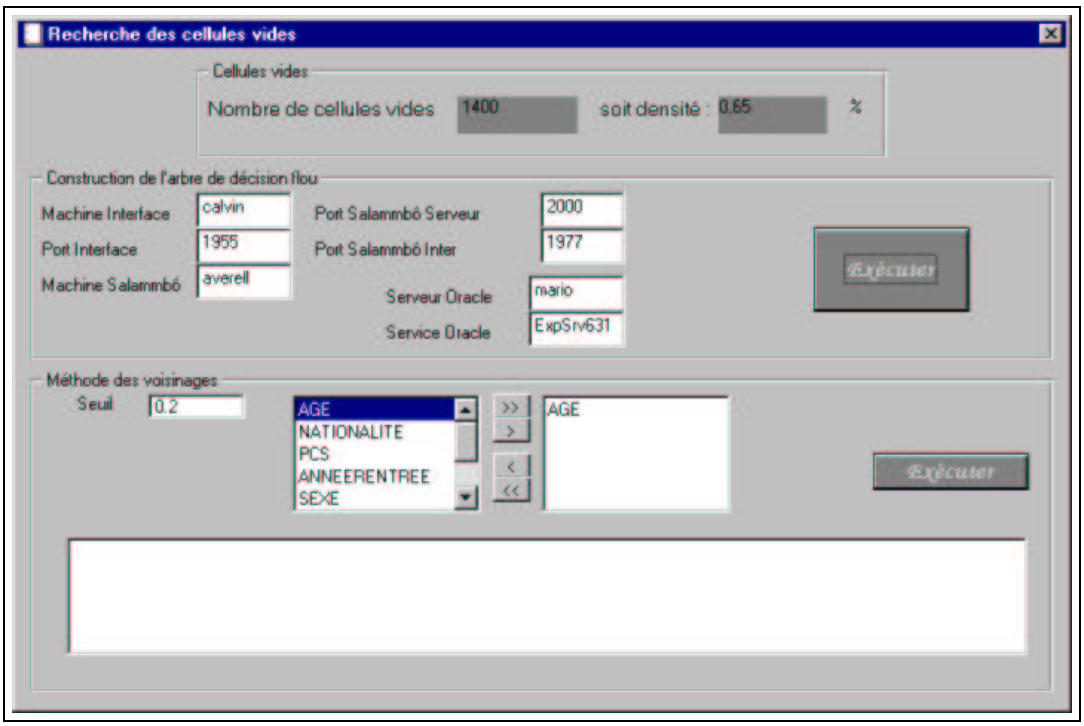


FIG. 57 – *FUB Miner* : recherche des cellules anormalement vides

## 5.1 Utilisation des arbres de décision flous

### 5.1.1 Mise en œuvre

Le module utilisant les arbres de décision ne nécessite aucune modification pour être appliqué à cette tâche. Seule une interface est développée afin de fournir les outils de construction du cube dont les cellules contiennent les deux classes à étudier : *vide* ou *non vide*. Ce cube est créé automatiquement par le système avec la valeur 0 pour les cellules vides et 1 pour les cellules non vides :

define TmpCube variable SHORTINTEGER < dims > ;
TmpCube = if (cube eq NA) then 0 else 1 ;

où dims est la liste des dimensions du cube à considérer.

Ce cube est effacé aussitôt après l'obtention du résultat.

L'arbre de décision est construit en fonction des choix de l'utilisateur sur les dimensions (comme décrit précédemment dans la présentation des arbres de décision, voir figure 47 de la page 156).

### 5.1.2 Résultats

Les tests ont été effectués sur la base $BAC$ pour construire l'arbre de décision à partir du cube $PAdm$. Nous insistons sur le fait que nous ne recherchons en aucun cas ici des données manquantes, la base BAC étant complète sans bruit. Il s'agit de retrouver les cellules anormalement vides dénotant des catégories de population (décrites selon les dimensions du cube) n'ayant pas passé un certain type de baccalauréat. Si l'on reprenait la base relationnelle source, il s'agirait de retrouver des lignes de la relation anormalement absentes, ce qui n'est pas possible avec les opérateurs relationnels existants.

À partir de l'arbre obtenu dans cet exemple, les résultats suivants sont extraits :

**R1** Au cours des deux années étudiées, il y a eu des garçons français de 15 ans admis dans les établissements privés sous contrat pour la série scientifique mais aucun résultat pour les filles n'est disponible sur les mêmes critères.

**R2** Au cours des deux années étudiées, il y a eu des filles françaises de 15 ans admises dans les établissements publics pour la série littéraire mais aucun résultat pour les garçons n'est disponible sur les mêmes critères.

## 5.2 Calcul des voisinages et des degrés d'intérêt

### 5.2.1 Mise en œuvre

En ce qui concerne les méthodes par voisinage, l'utilisateur choisit les dimensions sur lesquelles les voisinages doivent être construits. Le programme interne Express est ensuite généré *à la volée* selon les choix de l'utilisateur puis détruit après exécution. Les cellules résultats ainsi que leur degré d'intérêt sont listées.

Dans notre expérimentation, nous utilisons comme degré d'intérêt d'une cellule vide la proportion de cellules vides dans le voisinage. Plus ce degré est faible, plus la cellule est potentiellement intéressante (elle dénote au côté de nombreuses cellules non vides). On note que toutes les cellules vides d'un voisinage ont même degré d'intérêt.

L'utilisateur choisit le type de voisinage à utiliser ainsi que le seuil minimal du degré d'intérêt des cellules vides (qui constitue l'unique paramètre du programme). Ce choix est

effectué en sélectionnant les dimensions à prendre en compte, pour les dimensions possédant une hiérarchie. Un programme Oracle Express de type *Stored Procedure* est calculé à la volée en fonction du choix de ce voisinage.

### 5.2.2 Résultats

L'utilisation des méthodes décrites dans nos travaux (calcul des voisinages et de degrés d'intérêt) réduit considérablement le nombre de cellules vides que l'analyste doit considérer. Les résultats sont obtenus en quelques secondes.

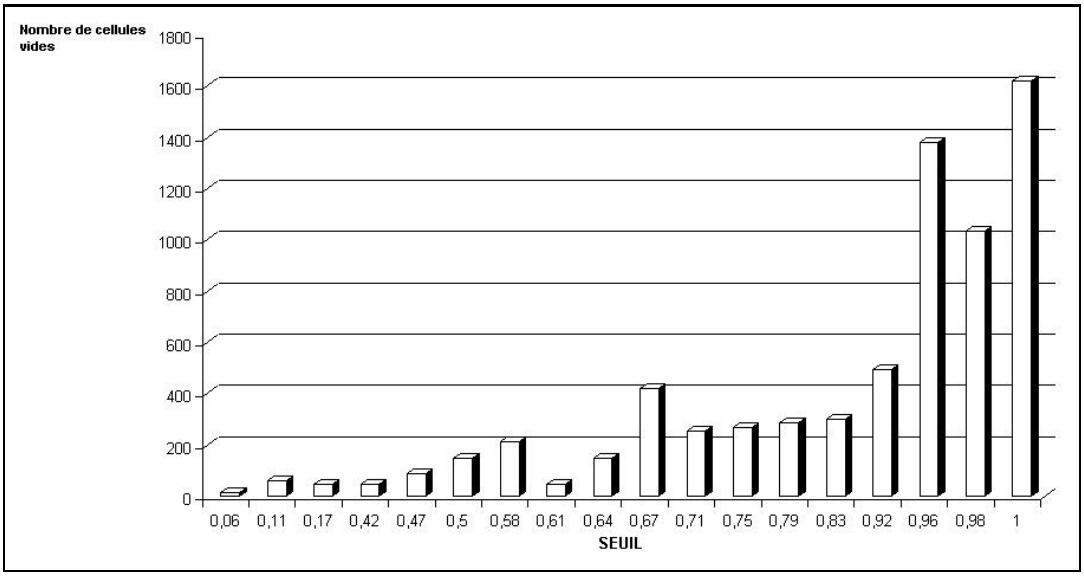


FIG. 58 – Nombre de cellules vides par degré d'intérêt

La figure 58 représente le nombre de cellules vides pour chaque proportion possible.

La figure 59 montre le nombre de cellules vides détectées comme potentiellement intéressantes pour chaque seuil de proportion possible.

Ces résultats concernent la base BACCALAUREAT, pour un cube défini sur huit dimensions dont cinq hiérarchisées. Le cube contient 9000 cellules dont 6869 vides. Les voisinages sont calculés par bloc, c'est-à-dire que les cellules sont regroupées si elles ont même parent sur l'ensemble des dimensions hiérarchisées du cube. Au seuil le plus faible, l'utilisateur n'a que 15 cellules à analyser. Le seuil 1 concerne les blocs de cellules vides, toutes les cellules vides s'y retrouvent donc. L'augmentation rapide du nombre de cellules pour les fortes proportions est due aux nombreux blocs vides ou quasi vides.

La taille du voisinage (nombre de cellules) varie de manière inverse par rapport au nombre de dimensions sur lesquelles il est calculé. La comparaison du nombre de cellules à étudier pour chacun des seuils de proportion maximal montre que plus le voisinage est restreint, plus le nombre de cellules vides à considérer est important.

La figure 60 représente le nombre de cellules vides selon chaque valeur de proportion de cellules vides pour trois types de voisinage. On note que le nombre de voisinages ne contenant que des cellules vides (proportion valant 1) diminue quand le nombre de dimensions prises en compte dans le calcul du voisinage augmente. Ceci s'explique par le fait que dans un voisinage de taille donnée, il existe plusieurs voisinages de taille inférieure.

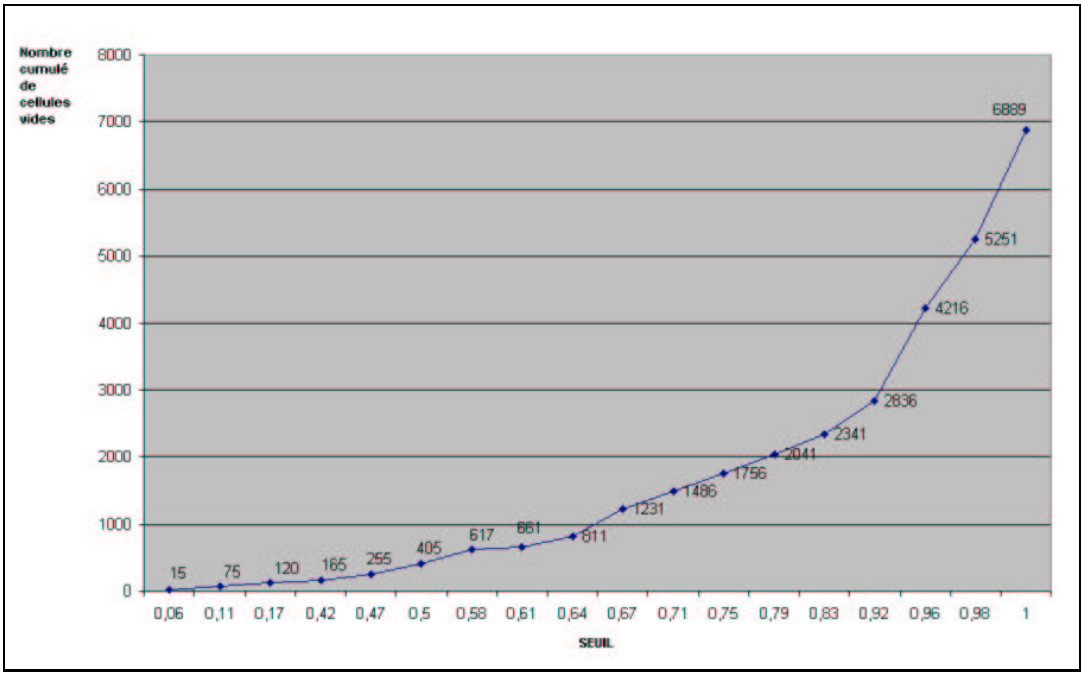


FIG. 59 – Nombre cumulé de cellules vides potentiellement intéressantes

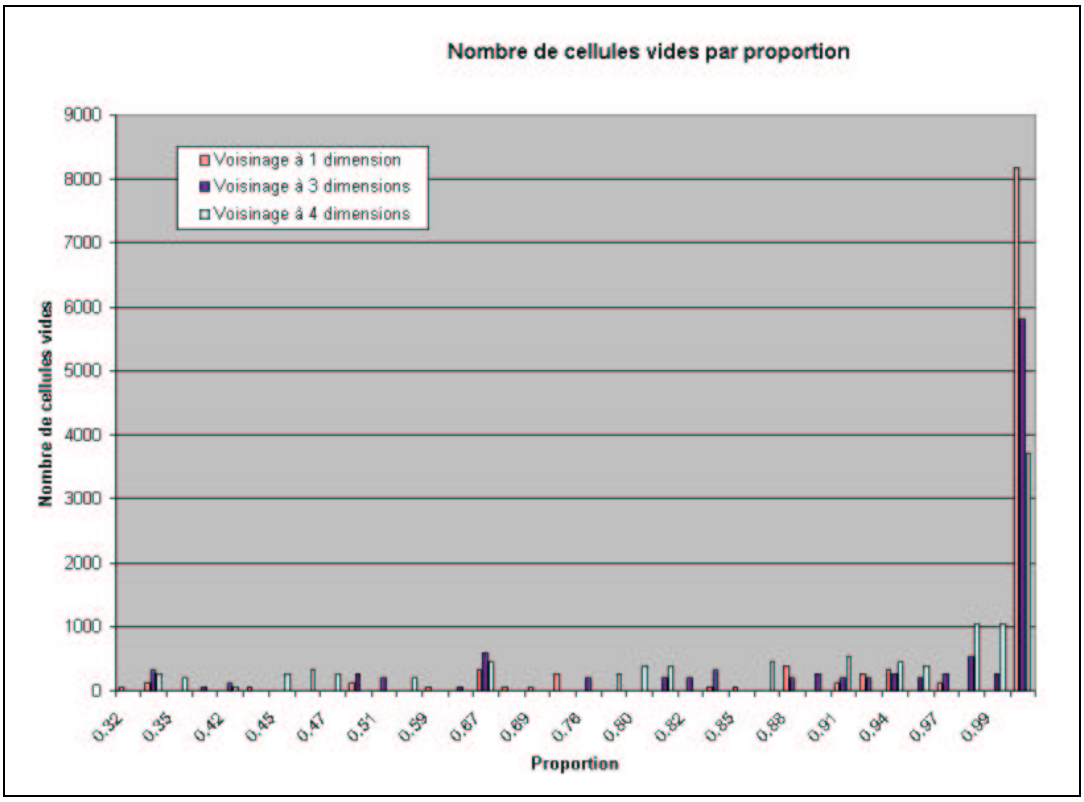


FIG. 60 – Nombre de cellules vides par degré d'intérêt pour chaque voisinage

La figure 61 illustre cette comparaison. Les courbes tracées correspondent à la variation du nombre de cellules potentiellement intéressantes en fonction de la proportion maximale de cellules vides dans le voisinage. Le cube utilisé compte 15840 cellules dont 10758 vides. La première courbe correspond à un voisinage de taille 1, c'est-à-dire calculé le long de la seule dimension $AGE$, la deuxième au voisinage calculé à partir des dimensions $AGE$, $PCS$ (catégorie socio-professionnelle des parents) et $STATUT$ (statut de l'établissement dans lequel le baccalauréat a été préparé, public ou privé), et la troisième au voisinage calculé à partir des dimensions $AGE$, $PCS$, $STATUT$ et $SERIEBAC$.

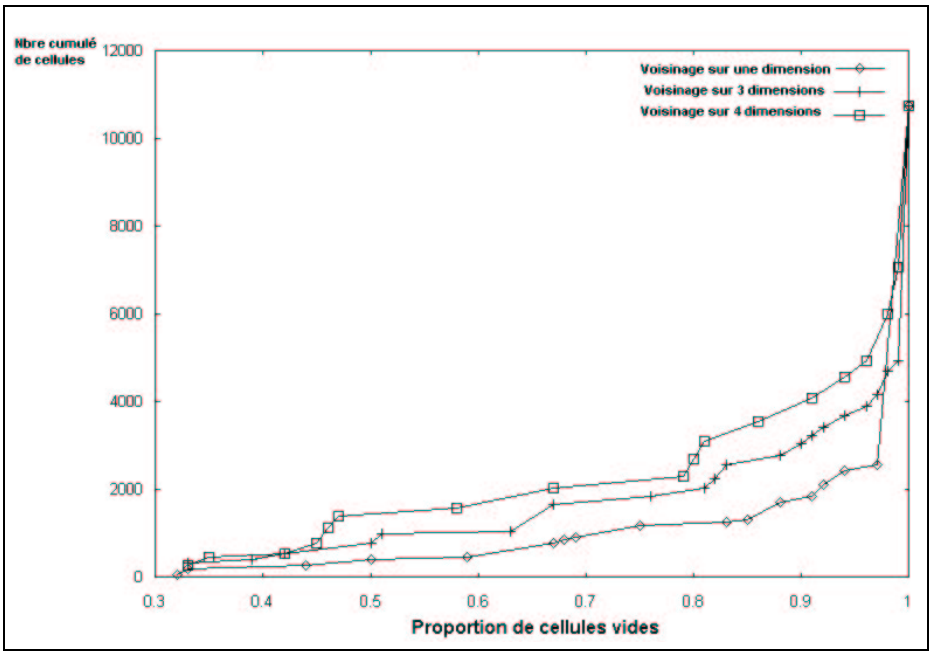


FIG. 61 – Augmentation du nombre de cellules vides selon la taille du voisinage

Ce résultat peut sembler étonnant, car dans le cas d'une distribution uniforme des cellules vides le long des voisinages, la proportion de cellules vides ne devrait pas varier avec le voisinage choisi. En réduisant le nombre de dimensions sur lequel le voisinage est calculé, le nombre de cellules vides est censé augmenter de manière similaire au nombre de cellules total.

# Conclusion et perspectives

## Contribution

Dans cette thèse, nous avons présenté nos travaux concernant la fouille de données à partir de bases de données multidimensionnelles floues. Ces travaux se situent à l'intersection de plusieurs disciplines : bases de données, fouille de données, logique floue et sciences cognitives.

L'approche OLAP et les bases de données multidimensionnelles, développées pour la gestion efficace de gros volumes de données à des fins d'analyse, s'est révélée particulièrement intéressante d'une part pour la visualisation des données à différents niveaux de granularité, mais aussi pour des analyses automatiques plus complexes avec les systèmes d'*OLAP Mining*.

De plus, l'utilisation du cadre formel fourni par la théorie des sous-ensembles flous permet la prise en compte de données imparfaitement connues et la génération de connaissance pertinente présentée à l'analyste en des termes du langage naturel et non pas sous la forme brute d'intervalles stricts de données numériques. Ces règles sont en outre mieux généralisables du fait de la gradualité des transitions entre les descriptions des données.

Enfin, le cadre des sciences cognitives permet notamment l'interprétation et la gestion efficace des problèmes liés à l'utilisation des sous-ensembles flous. Dans notre travail, une étude est menée afin de mieux connaître les méthodes employées de manière naturelle pour résumer des données numériques et pour mieux cerner l'utilisation des quantificateurs flous du type *la plupart*.

Cependant, aucun système intégrant le flou dans les outils OLAP n'existe dans la littérature, ni pour la représentation et la manipulation des données, ni pour leur analyse. L'approche présentée est donc novatrice.

Nous définissons un modèle de représentation de données potentiellement imparfaites par l'extension des définitions de toutes les entités classiques des bases de données multidimensionnelles : éléments de cellule, hiérarchies, dimensions et domaines de dimensions, et cubes flous.

Les cubes flous sont manipulés à l'aide des opérateurs classiques des bases de données multidimensionnelles étendus pour la prise en compte de valeurs imparfaites et de critères vagues : sélection sur les cellules et sur les tranches, agrégation et généralisation, projection, ainsi que les opérateurs binaires de combinaison de cubes de manière conjonctive ou disjonctive.

Notre proposition est intégrée dans un système d'OLAP Mining flou, où plusieurs processus de fouille de données sont utilisés sur les bases de données multidimensionnelles floues. Trois modules sont disponibles :

- construction d'arbres de décision flous,
- génération de résumés flous,
- recherche des cellules anormalement vides.

Ce modèle, ainsi que le système modulaire de fouille de données ont été implantés dans les deux systèmes prototypes :

- *FUB* pour la représentation, la visualisation et la manipulation de bases de données multidimensionnelles floues,
- *FUB Miner* pour l'analyse de ces bases de données à l'aide des trois modules de fouille de données présentés ci-dessus (construction d'arbres de décision flous, génération de résumés flous et recherche des cellules anormalement vides).

Dans les deux systèmes, des interfaces conviviales sont disponibles pour guider l'analyste à travers les différentes fonctionnalités qui lui sont proposées.

Des tests ont été effectués sur deux bases de données. La première est une base fournie avec Oracle Express décrivant des résultats de ventes de produits dans des villes américaines. La seconde a été fournie par le Ministère de l'Éducation Nationale et décrit les résultats au baccalauréat sur deux années consécutives. Dans cette application en particulier, l'utilisation de bases multidimensionnelles garantit la conservation de l'anonymat des élèves en travaillant à un niveau agrégé après la construction du cube.

Les résultats obtenus montrent l'intérêt de notre approche, et la pertinence des règles obtenues.

# Perspectives

Les perspectives associées à ce travail sont nombreuses, et concernent différents aspects du modèle, de son implantation, et du système de fouille de données.

Nous décrivons ci-dessous l'ensemble des perspectives liées au travail amorcé depuis le début de cette thèse. Ces perspectives sont distinguées selon qu'elles concernent le modèle de représentation et de manipulation des données imparfaites, ou le système de fouille de données à partir de bases de données multidimensionnelles, ou encore les aspects cognitifs liés à ce travail.

## Modèle multidimensionnel flou

### Traduction du modèle en ROLAP

L'intérêt de l'approche ROLAP réside dans la gestion de plus gros volumes de données que ceux déjà possibles avec une approche MOLAP comme celle proposée ici. Cependant, la traduction du modèle MOLAP en modèle ROLAP pose certains problèmes qui sont notamment les suivants :

- représentation des degrés de certitude et d'appartenance,
- traduction des opérations (sélection, projection, roll-up), cette question étant déjà posée dans le cas classique,
- extension de la définition de l'opérateur de construction de cubes (*group-by*/*cube by*).

### Fondements théoriques des bases de données multidimensionnelles floues

Ce point vise à l'approfondissement des problèmes théoriques liés à l'utilisation du modèle multidimensionnel et à son extension au flou :

- schémas de données pour entrepôts de données et bases décisionnelles après extension au flou,
- aspects sémantiques du modèle multidimensionnel flou,
- dépendances fonctionnelles et formes normales dans le modèle multidimensionnel flou.

### Construction des cubes

Nous envisageons l'intégration dans le modèle d'un module de construction des bases de données multidimensionnelles, et notamment des cubes, à partir de bases de données relationnelles, ou simplement stockées dans des fichiers.

Ce module devra donc intégrer des interfaces utilisateur, et des automatisations de construction. Le choix des cubes à construire est un problème difficile, comme nous l'avons présenté dans la section 2.6.

### Requêtes *What If* floues

Les requêtes *What If* sont de la forme *"Que se passerait-il si l'on augmentait le prix de vente de 20% ?"*. Le but ici serait d'étendre de telles requêtes au traitement de critères vagues de la forme *"Que se passerait-il si l'on augmentait* fortement *le prix de vente ?"*.

### Visualisation des cubes flous

Nous envisageons une méthode de visualisation des cubes pour bien distinguer ce qui appartient fortement ou faiblement au cube. La méthode envisagée est l'utilisation des couleurs, avec par exemple des couleurs graduelles plus ou moins foncées selon que telle cellule ou telle tranche appartient plus ou moins fortement au cube.

## Système de fouille de données multidimensionnelles floues (F-OLAP Mining)

### Couplage avec SODAS

SODAS (*Symbolic Official Data Analysis System*) est un logiciel (développé par plusieurs équipes au sein d'un projet européen) dédié à l'analyse d'*objets symboliques,* c'est-à-dire des données munies de variation [DID 99]. Les données ne sont plus traitées au niveau individuel mais à des niveaux de concept en formant pour tous les objets appartenant à un même concept des objets symboliques (par exemple des histogrammes). Les liens entre cette approche et la nôtre sont très forts, nous envisageons donc de faire collaborer les logiciels sur les points suivants :

- utilisation de *FUB* pour la production de fichiers SODAS après visualisation multidimensionnelle des données,
- utilisation de SODAS pour la construction de cubes à mesure complexe (objet symbolique) en utilisant l'outil DB2SO afin de traduire les données sources en données agrégées sous la forme d'objets symboliques intégrables dans un cube de données.

### Degré d'inclusion de l'algorithme d'apprentissage dans le SGBDM

Nous souhaitons étudier ici l'influence du degré d'inclusion de l'algorithme d'apprentissage dans le système de gestion de bases de données multidimensionnel sur les performances du système (sous l'hypothèse que le SGBDM est optimisé pour les calculs d'agrégats). On étudie ici le couplage des algorithmes d'apprentissage et du système de gestion de base de données, l'évolution des performances en fonction du degré d'intégration, et les limites de cette intégration de l'algorithme de fouille de données au SGBDM. Les pré-calculs (consolidations) possibles seront étudiés.

### Prise en compte des résultats de classification pour le choix des cubes à construire

Le but est d'améliorer de manière incrémentale les cubes servant à la fouille de données. On se place donc dans un contexte où les connaissances produites par la phase d'apprentissage sont réinjectées au début du processus pour la construction de cubes intéressants. Ces

connaissances sont alors traitées comme des méta-connaissances des données à analyser, au même titre que le schéma de la base relationnelle par exemple.

### Injection de nouveaux algorithmes d'apprentissage

Nous envisageons d'enrichir notre système de fouille de données par l'ajout de nouveaux modules.

- *Découverte d'exceptions.* La plupart des systèmes d'apprentissage classiques considèrent le cas général. Pourtant, la découverte des exceptions peut se révéler très intéressante et très pertinente. Différentes méthodes de découverte des exceptions existent. En considérant les algorithmes classiques d'apprentissage, on peut par exemple en déduire très facilement les exemples qui ne satisfont pas les règles trouvées. Par exemple, on peut repérer les exemples ne satisfaisant pas un arbre, les *outliers* des SVM, ou encore les exemples ne faisant partie d'aucun cluster.
  De nombreux travaux existent à ce sujet, qui seront étudiés pour leur éventuelle intégration dans notre système [SUZ 98, LIU 99, HUS 00].
- *Prototypes.* On envisage d'enrichir le système pour la génération de prototypes flous, en intégrant le système SQUAW de [RIF 96, RIF 98].

### Utilisation des cubes flous en data mining multimédia

Sur ce point, nos travaux reposent sur les recherches initiées par J. Han qui a utilisé un modèle multidimensionnel dans le cadre de bases multimédia [ZAI 98a, ZAI 98b, ZAI 00]. En outre, [DET 00] a étudié l'application de degrés d'incertitude et de leur agrégation dans le cadre de telles bases. On se propose donc d'utiliser ces travaux dans un processus d'OLAP Mining multimédia flou afin de profiter au mieux des avantages de chacune des méthodes.

## Sciences cognitives

### Etude des aspects cognitifs liés à la représentation multidimensionnelle

Le modèle multidimensionnel a été introduit par les analystes et les sociétés de production d'outils de visualisation et d'analyse avant même une étude plus *académique*, et a ensuite été repris dans le cadre de recherches plus *fondamentales*. On se propose donc d'étudier les propriétés qui font de ce modèle un très bon outil de visualisation et d'analyse. Ces propriétés sont essentiellement liées à la nature multidimensionnelle de la visualisation, à l'introduction d'opérations telles que les rotations ou les inversions de l'ordre des valeurs des domaines de dimension qui induisent parfois des proximités d'objets qui rendent plus aisée la découverte de propriétés sur les données, mais surtout à la possibilité offerte pour naviguer à travers différents niveaux de granularité.

### Problématiques liées à la pertinence :

- Phase d'apprentissage pour savoir ce qui est pertinent pour l'utilisateur/la classe d'utilisateur et par rapport à la tâche On se propose ici d'apprendre au cours des utilisations des profils d'utilisateurs afin de cerner ce qui est pertinent selon les utilisateurs et de générer une information pertinente pour chaque utilisateur et chaque type d'utilisation.

- *Comparaison de la pertinence de paquets de règles* (lequel est le plus pertinent pour une classe d'individus donnée ?). On souhaite comparer ici la pertinence des connaissances produites par rapport à l'utilisateur.
- *Critères de pertinence.* Le but est de recenser tous les éléments influant sur la pertinence des connaissances produites (par exemple longueur des règles/précision, niveau de généralité) pour être capable d'évaluer de manière automatique les pertinences associées à chaque découverte.

# Bibliographie


[AGR 93] Agrawal R., Imielinski T., Swami A., "Mining Association Rules between Sets of Items in Large Databases", Buneman P., Jajodia S., Eds., *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, Washington, D.C., p. 207–216, 1993.

[AGR 97] Agrawal R., Gupta A., Sarawagi S., "Modeling Multidimensional Databases", Gray A., Larson P.-Å., Eds., *Proc. 13th Int. Conf. Data Engineering, ICDE*, Birmingham U.K., IEEE Computer Society, p. 232–243, 1997.

[ALA 97] Aladenise N., Bouchon-Meunier B., "Acquisition de connaissances imparfaites : mise en évidence d'une fonction d'appartenance", *Revue internationale de sytémique*, vol. 11, n° 1, p. 109–127, 1997.

[ALB 99a] Albanesi M., Ferreti M., Giancane A., "A Visual Tool for fuzzy Data Analysis and Decision Support", *Proc. of the International Conference on Information Systems, Analysis and Synthesis - World Multiconference on Systemics, Cybernetics and Informatics (SCI'99)*, vol. 2, 1999.

[ALB 99b] Albrecht J., Bauer A., Deyerling O., Guenzel H., Huemmer W., Lehner W., Schlesinger L., "Management of multidimensional Aggregates for efficient Online Analytical Processing", *Proc. of the Int. Database Engineering and Applications Symposium (IDEAS)*, Montreal, Canada, p. 156–164, 1999.

[AND 95] Andreasen T., Pivert O., "Un mécanisme coopératif d'affaiblissement pour les requêtes relationnelles floues", *Rencontres francophones sur la logique floue et ses applications*, Paris, France, Cépaduès Editions, p. 134–139, 1995.

[BAR 97] Baralis E., Paraboschi S., Teniente E., "Materialized Views Selection in a Multidimensional Database", Jarke M., Carey M., Dittrich K., Lochovsky F., Loucopoulos P., Jeusfeld M., Eds., *VLDB'97, Proceedings of 23rd International Conference on Very Large Data Bases, August 25-29, 1997, Athens, Greece*, Morgan Kaufmann, p. 156-165, 1997.

[BAS 02] Bastide Y., Taouil R., Pasquier N., Stumme G., Lakhal L., "Pascal, un algorithme d'extraction des motifs frequents", *Technique et Science Informatiques (TSI)*, vol. 21, n° 1, p. 65–95, Hermes, 2002.

[BAT 82] Bates D., Boral H., DeWitt D., "A Framework for Research in Database Management for Statistical Analysis", Schkolnick M., Ed., *Proceedings of the 1982 ACM SIGMOD International Conference on Management of Data, Orlando, Florida, June 2-4, 1982*, ACM Press, p. 69-78, 1982.

[BEN 93] Benzaken V., Doucet A., *Bases de données orientées objets : origines et principes*, Collection Acquis Avancés de l'Informatique, Armand-Colin, 1993.

[BEY 99] Beyer K., Ramakrishnan R., "Bottom-Up Computation of Sparse and Iceberg CUBEs", Delis A., Faloutsos C., Ghandeharizadeh S., Eds., *SIGMOD 1999, Proceedings ACM SIGMOD International Conference on Management of Data, June 1-3, 1999, Philadephia, Pennsylvania, USA*, ACM Press, p. 359–370, 1999.

[BIL 96] Bilgiç T., Turksen I. B., "Measurement of membership functions : a survey", 1996, transparents de la communication effectuée lors du "Canadian-Japanese bilateral workshop on Fuzzy Sets and Neural Integration", Toronto, disponibles sur internet http ://www.ie.boun.edu.tr/~taner.

[BIL 99] Bilgiç T., Turksen I. B., "Measurement of Membership Functions : Theoretical and Empirical Work", Dubois D., Prade H., Eds., *Handbook of Fuzzy Sets and Systems, Fundamentals of Fuzzy Sets*, vol. 1, Chapitre 3, p. 195–232, Kluwer, 1999.

[BLA 98] Blaschka M., Sapia C., Hofling G., Dinter B., "Finding your way through Multidimensional Data Models", *Proc. International Workshop on Data Warehouse Design and OLAP Technology (DWDOT, in connection with DEXA)*, Vienna, Austria, p. 198–203, 1998.

[BOA 01] Boaretto Y., "La Caisse des Dépôts ouvre un coffre-fort numérique", *VNUnet, http ://www.vnunet.fr/actu/article.htm ?numero=7544*, VNU Publications France, 3 mai 2001.

[BON 95] Bonano N., Mouaddib N., "Sémantique de degré d'appartenance d'un tuple à une relation : Discussion et Propositions", *Rencontres francophones sur la logique floue et ses applications*, Paris, France, Cépaduès Editions, p. 140–147, 1995.

[BOS 92] Bosc P., Pivert O., "Fuzzy Querying in conventional databases", Zadeh L., Kacprzyk J., Eds., *Fuzzy Logic for the Management of Uncertainty*, Chapitre 32, p. 645–671, John Wisley, 1992.

[BOS 95a] Bosc P., Kacprzyk J., Eds., *Fuzziness in Database Management Systems, Studies in fuzziness*, Springer-Verlag, 1995.

[BOS 95b] Bosc P., Pivert O., "SQLf : A relational database language for fuzzy querying", *IEEE Transactions on Fuzzy Systems*, vol. 3, p. 1–17, 1995.

[BOS 97a] Bosc P., Pivert O., "On the Comparison of Imprecise Values in Fuzzy Datbases", *Proc. Sixth IEEE Int'l Conf. on Fuzzy Systems. (FUZZ-IEEE '97)*, Barcelona, Spain, IEEE Press, p. 707–712, 1997.

[BOS 97b] Bosc P., Prade H., "An introduction to the fuzzy set and possibility theory-based treatment of soft queries and uncertain or imprecise databases", Motro A., Smets P., Eds., *Uncertainty and Management in Information Systems : From Needs to Solutions*, p. 285–324, Kluwer Academic Publishers, 1997.

[BOS 98a] Bosc P., Dubois D., Prade H., "Fuzzy Functional Dependencies -An Overview and a critical discussion", *Journal of the American Society for Information Science*, vol. 49, p. 217–235, 1998.

[BOS 98b] Bosc P., Liétard L., "Functional Dependencies and the Design of relational databases extended to Imprecise Data", *7th Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU)*, Université de La Sorbonne, Paris, France, EDK Editions, p. 412–419, 1998.

[BOS 98c] Bosc P., Liétard L., Pivert O., "Extended Functional Dependencies as a Basis for Linguistic Summaries", *2nd European Symposium on Principles of Data Mining*

*and Knowledge Discovery (PKDD)*, vol. 1510 de *Lecture Notes in Artificial Intelligence*, Nantes, France, Springer Verlag, p. 93-101, 1998.

[BOS 98d] Bosc P., Pivert O., "Fuzzy Databases", Bonissone P., Pedrycz W., Eds., *Handbook of Fuzzy Computation*, Chapitre 4.1, p. 1–12, Oxford University Press, 1998.

[BOS 99] Bosc P., Pivert O., Ughetto L., "On Data Summaries Based on Gradual Rules", Reusch B., Ed., *Fuzzy Days*, vol. 1625 de *Lecture Notes in Computer Science*, Springer, p. 512–521, 1999.

[BOS 00a] Bosc P., Dubois D., Pivert O., Prade H., "Résumés de données et ensembles flous - Principe d'une nouvelle approche", *Rencontres francophones sur la logique floue et ses applications (LFA)*, La Rochelle, France, Cépaduès Editions, p. 333-340, 2000.

[BOS 00b] Bosc P., Liétard L., "Cardinalités floues et Interrogation flexible de bases de données", *Rencontres Francophones sur la Logique Floue et ses Applications*, La Rochelle, France, Cepadues Editions, p. 281–288, 2000.

[BOS 00c] Bosc P., Pivert O., "Deux types de sémantique pour les requêtes adressées à des bases de données possibilistes", *Rencontres Francophones sur la Logique Floue et ses Applications*, La Rochelle, France, Cepadues Editions, p. 289–296, 2000.

[BOS 01] Bosc P., Duval L., Pivert O., "A First Approach to Possibilistic Queries Addressed to Possibilistic Databases", *Proc. of the IFSA / NAFIPS 2001*, Vancouver, Canada, p. 2452–2457, 2001.

[BOS 02] Bosc P., Pivert O., Liétard L., "On the Comparison of aggregates over fuzzy sets", *Proceedings of the IEEE International Conference on Fuzzy Systems (FUZZ-IEEE'02)*, Honolulu, Hawaii, 2002.

[BOU 94] Bouchon-Meunier B., *La logique floue*, PUF, 1994.

[BOU 95] Bouchon-Meunier B., *La logique floue et ses applications*, Addison Wesley, Paris, 1995.

[BOU 96] Bouchon-Meunier B., Rifqi M., Bothorel S., "Towards general Measures of Comparison of Objects", *Fuzzy Sets and Systems*, vol. 84, p. 143–153, Elsevier Science, 1996.

[BOU 97] Bouchon-Meunier B., Marsala C., Ramdani M., "Learning from Imperfect Data", Dubois D., Prade H., Yager R. R., Eds., *Fuzzy Information Engineering : a Guided Tour of Applications*, Chapitre 8, p. 139-148, John Wileys and Sons publisher, 1997.

[BOU 99a] Bouchon-Meunier B., Marsala C., "Learning Fuzzy Decision Rules", Bezdek J., Dubois D., Prade H., Eds., *Fuzzy Sets in Approximate Reasoning and Information Systems, The Handbooks on Fuzzy Sets Series*, Chapitre 4, p. 279-304, Kluwer Academic, 1999.

[BOU 99b] Boulicaut J., Marcel P., Rigotti C., "Query Driven Knowledge Discovery in Multidimensional Data", *ACM Second International Workshop on Data Warehousing and OLAP (DOLAP'99)*, Kansas City (USA), ACM Press, p. 87-93, November 6 1999.

[BOU 00a] Boulicaut J.-F., "Quelques problèmes ouverts pour la découverte de règles dans les bases de données (conférence invitée)", *Actes des Rencontres Francophones sur la logique floue et ses applications (LFA'00)*, La Rochelle, France, Cepadues Editions, p. 357–361, octobre 2000.

[BOU 00b] Boulicaut J.-F., Bykowski A., Rigotti C., "Approximation of frequency queries by mean of free-sets", *Proceedings of the Fourth European Conference on Principles and Practice of Knowledge Discovery in Databases PKDD'00*, vol. 1910 de *Lecture Notes in Artificial Intelligence*, Lyon, France, Springer-Verlag, p. 75–85, 2000.

[BOU 00c] Boulicaut J.-F., Bykowski A., Rigotti C., "Free-sets : a condensed representation of boolean data for frequency query approximation", Accepté pour publication dans *Data Mining and Knowledge Discovery journal*, Kluwer Academic Publishers, 2000.

[BOU 00d] Boulicaut J.-F., Jeudy B., "Using constraint for itemset mining : should we prune or not ?", Doucet A., Ed., *Proceedings Bases de Données Avançées (BDA'00)*, Blois, F, p. 221–237, 2000.

[BRA 81] Bragg A., "Data Manipulation Languages for Statistical Databases - the Statistical Analysis System (SAS)", Wong H., Ed., *Proceedings of the First LBL Workshop on Statistical Database Management, Melno Park, California, USA, December 2-4, 1981*, Lawrence Berkeley Laboratory, p. 147-150, 1981.

[BRE 84] Breiman L., Friedman J., Olshen R., Stone C., *Classification And Regression Trees*, Chapman and Hall, New York, 1984.

[BUC 93] Buckles B., Petry F., "A fuzzy representation of data for relational databases", Dubois D., Prade H., Yager R., Eds., *Readings in Fuzzy Sets for Intelligent Systems*, p. 660–666, Morgan Kaufmann Publisher, 1993.

[CAB 97] Cabibbo L., Torlone R., "Querying multidimensional databases", *Sixth Int. Workshop on Database Programming Languages*, Estes Park, Colorado, USA, p. 253–269, 1997.

[CAB 98] Cabibbo L., Torlone R., "A Logical Approach to Multidimensional Databases", *Sixth International Conference on Extending Database Technology (EDBT'98)*, Valencia, Spain, Lecture Notes in Computer Science 1377,Springer-Verlag, p. 183-197, 1998.

[CHA 81] Chan P., Shoshani A., "SUBJECT : A Directory Driven System for Large Statistical Databases", Wong H. K. T., Ed., *Proceedings of the First LBL Workshop on Statistical Database Management*, Melno Park, California, USA, Lawrence Berkeley Laboratory, p. 61-62, 1981.

[CHA 83] Chan P., Eggers S., Gey F., Holmes H., Kreps P., McCarthy J., Merrill D., Olken F., Shoshani A., Wong H., "Statistical Data Management Research at Lawrence Berkeley Laboratory", Hammond R., McCarthy J., Eds., *Proceedings of the Second International Workshop on Statistical Database Management, Los Altos, California, USA*, Lawrence Berkeley Laboratory, p. 273-279, 1983.

[CHA 94] Chaudhry N., Moyne J., Rundensteiner E., "A Design Methodology for Databases with Uncertain Data", French J., Hinterberger H., Eds., $7^{th}$ *International conference on Scientific and Statistical Database Management*, Charlottesville, Virginia, p. 32–41, 1994.

[CHA 97] Chaudhuri S., Dayal U., "An overview of Data Warehousing and OLAP Technology", *ACM SIGMOD record*, vol. 26, n° 1, p. 517–526, March 1997.

[CHA 98] Chaudhuri S., "Data Mining and Database Systems : Where is the Intersection ?", *Data Engineering Bulletin*, vol. 21, n° 1, p. 4-8, 1998.

[CHE 95] Chen G., "Fuzzy Functional Dependency and A series of Design Issues of fuzzy relational databases", *Fuzziness in Database Management Systems, Studies in fuzziness*, p. 166–185, Springer-Verlag, 1995.

[CHE 96] Chen M., Han J., Yu P., "Data Mining : An Overview from a Database Perspective", *IEEE Transactions on Knowledge and Data Engineering*, vol. 8, n° 6, p. 866-883, 1996.

[CHO 01] Choong Y., Laurent D., Marcel P., "Computing Appropriate Representations for Multidimensional Data", *Proc. ACM 4th International Workshop on Data Warehousing and OLAP (DOLAP'01)*, Atlanta, Georgia, USA, ACM Press, 2001.

[CIC 01] Cicchetti R., Novelli N., Lakhal L., "APIC : An efficient Algorithm for Computing Iceberg Datacubes", *Actes de la 17ème Conférence Bases de Données Avancées (BDA'01)*, Agadir, Maroc, Cepadues-Edition, p. 229–242, 2001.

[COD 70] Codd E., "A relational model of data for large shared data banks", *Communications of the ACM*, vol. 13, p. 377–387, 1970.

[COD 93] Codd E., Codd S., Salley C., Providing OLAP (On-Line Analytical Processing) to User-Analysts : An IT Mandate, Report , Arbor Software White Paper, 1993.

[CUB 94] Cubero J., Vila M., "A new definition of fuzzy functional dependency in fuzzy relational databases", *Journal of Intelligent Systems*, vol. 9, n° 5, p. 441–448, 1994.

[CUB 97] Cubero J., Medina J., Pons O., Vila. M., "Data Summarization with Linguistic Labels : A Loss Less Decomposition Approach", *Proc. International Fuzzy Systems Association World Congress (IFSA)*, Prague, 1997.

[DAT 99] Datta A., Thomas H., "The Cube Data Model : A Conceptual Model and Algebra for On-Line Analytical Processing in Data Warehouses", *Decision Support Systems*, vol. 27, n° 3, p. 289–301, 1999.

[DEH 01] Dehne F., Eavis T., Hambrusch S., Rau-Chaplin A., "Parallelizing The Data Cube", *Journal on Distributed and Parallel Databases (Special Issue on Parallel and Distributed Data Mining)*, vol. 11, n° 2, p. 181-201, Kluwer Academic, 2001.

[DEL 91] Delobel C., Lécluse C., Richard P., *Bases de données : des systèmes relationnels aux systèmes à objets*, InterEditions, 1991.

[DEN 83] Denning D., Nicholson W., Sande G., Shoshani A., "Research Topics in Statistical Database Management", Hammond R., McCarthy J., Eds., *Proceedings of the Second International Workshop on Statistical Database Management, Los Altos, California, USA*, Lawrence Berkeley Laboratory, p. 46-51, 1983.

[DER 98] Deret D., "Le syllogisme catégorique : une réexamination du terme "certains" dans une perspective développementale", *Canadian Journal of Experimental Psychology*, vol. 52, n° 4, 1998.

[DES 97] Deshpande P., Naughton J., Ramasamy K., Shukla A., Tufte K., Zhao Y., "Cubing Algorithms, Storage Estimation, and Storage and Processing Alternatives for OLAP", *Data Engineering Bulletin*, vol. 20, n° 1, p. 3-11, 1997.

[DET 00] Detyniecki M., Mathematical Aggregation Operators and their Application to Video Querying, PhD thesis, Université Paris 6, 2000.

[DID 82] Diday E., *Eléments d'analyse de données*, Dunod, 1982.

[DID 99] Diday E., "An introduction to symbolic data analysis and its application to the SODAS project"", *Cahiers du CEREMADE, N° 9914*, 1999.

[DUB 90] Dubois D., Prade H., "Measuring properties of fuzzy sets : a general technique and its use in fuzzy query evaluation", *Fuzzy Sets and Systems*, vol. 38, p. 137–152, 1990.

[DUB 97a] DUBOIS D., PRADE H., "Using fuzzy sets in flexible querying : Why and how ?", *Flexible Query Answering Systems (FQAS)*, Kluwer Academic Publishers, p. 45–60, 1997.

[DUB 97b] DUBOIS D., PRADE H., RANNOU E., "User-Driven Summarization of Data Based on Gradual Rules", *Int. conf. on Fuzzy systems, FUZZ'IEEE*, vol. 2, Barcelona, Spain, p. 839-844, 1997.

[DUB 00] DUBOIS D., PRADE H., "Fuzzy sets in data summaries - Outline of a new approach", *Proc. of the 8th Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU)*, Madrid, Espagne, p. 1035-1040, 2000.

[FAN 98] FANG M., SHIVAKUMAR N., GARCIA-MOLINA H., MOTWANI R., ULLMAN J., "Computing Iceberg Queries Efficiently", *Proc. 24th Int. Conf. Very Large Data Bases, VLDB*, New York, USA, p. 299–310, 24–27 1998.

[FEN 99] FENG L., DILLON T., "Enhancing Data Warehousing with Fuzzy Technology", *Proceedings of the 10th Int. Conf. DEXA'99*, vol. 1677 de *Lecture Notes in Computer Science*, Springer Verlag, p. 872–881, 1999.

[FIN 00] FINDLATER L., HAMILTON H., An Empirical Comparison of Methods for Iceberg-CUBE Construction, Report n° CS-2000-06, University of Regina, CANADA, 2000.

[FIR 97] FIRESTONE J. M., "Evaluating OLAP Alternatives", *White Paper No. 4*, Executive Information Systems, Inc., 1997.

[GHO 88] GHOSH S., "Statistical Relational Model", RAFANELLI M., KLENSIN J., SVENSSON P., Eds., *Statistical and Scientific Database Management, 4th International Working Conference SSDBM, Proceedings*, vol. 339 de *Lecture Notes in Computer Science*, Rome, Italy, Springer, p. 338-355, 1988.

[GHO 91] GHOSH S., "Statistical Relational Databases : Normal Forms", *IEEE Transactions on Knowledge and Data Engineering*, vol. 3, n° 1, p. 55-64, 1991.

[GOI 98] GOIL S., CHOUDHARY A., "High Performance Multidimensional Analysis and Data Mining", *Proc. SC98 : High Performance Networking and Computing Conference*, Orlando, USA, 1998.

[GRA 97] GRAY J., BOSWORTH A., LAYMAN A., PIRAHESH H., "Data Cube : A Relational Aggregation Operator Generalizing Group-By, Cross-Tabs, and Sub-Totals", *Journal of Data Mining and Knowledge Discovery*, vol. 1, n° 1, p. 29–53, Kluwer Academic Publishers, 1997.

[GUE 99] GUENZEL H., ALBRECHT J., LEHNER W., "Data Mining in a Multidimensional Environment", *Proc. 3th East-European Symposium on Advances in Databases and Information Systems (ADBIS'99,Maribor,Slovenia, Sept. 13 - 16)*, 1999.

[GUE 00] GUENZEL H., ALBRECHT J., LEHNER W., "Use and Reuse of Association Rules in an OLAP Environment", *Proceedings of IRMA 2000 (2000 Information Resources Management Association International Conference)*, Anchorage, Alaska, May 2000.

[GYS 97] GYSSENS M., LAKSHMANAN L., "A Foundation for Multidimensional Databases", *Proc. 23rd Int. Conf. on Very Large Data Bases*, Athens, Greece, p. 106–115, August 1997.

[HAL 95] HALE J., SHENOI S., "Catalyzing Database Inference with Fuzzy Relations", *Proceedings of the Third International Symposium on Uncertainty Modeling and Analysis*, Maryland, USA, IEEE Computer Society, 1995.

[HAN 96] Han J., Fu Y., Wang W., Chiang J., Gong W., Koperski K., Li D., Lu Y., Rajan A., Stefanovic N., Xia B., Zaiane O., "DBMiner : A System for mining knowledge in relational databases", Simoudis E., Han J., Fayyad U., Eds., *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD'96)*, Portland, Oregon, AAAI Press, p. 250–255, 1996.

[HAN 97] Han J., "OLAP Mining : An Integration of OLAP with Data Mining", *IFIP Conference on Data Semantics*, p. 1-11, 1997.

[HAN 98] Han J., "Towards On-Line Analytical Mining in Large Databases", *SIGMOD Record (ACM Special Interest Group on Management of Data)*, vol. 27, n° 1, p. 97–107, 1998.

[HAN 99] Hanss M., "The implementation of fuzzy arithmetical operations for engineering problems", *Proc. of the NAFIPS*, New York, USA, p. 462–466, 1999.

[HAN 01] Han J., Pei J., Dong G., Wang K., "Efficient Computation of Iceberg Cubes with Complex Measures", *ACM-SIGMOD Int. Conf. on Management of Data (SIGMOD'01)*, Santa Barbara, USA, 2001.

[HAR 96] Harinarayan V., Rajaraman A., Ullman J., "Implementing data cubes efficiently", *ACM SIGMOD International Conference on Management of Data*, Montreal, Canada, p. 205–216, 1996.

[HOL ] Holos, Crystal Decisions, http ://www.crystaldecisions.com/products/holos.

[HUS 00] Hussain F., Liu H., Suzuki E., Lu H., "Exception Rule Mining with a Relative Interstingness Measure", *PAKDD*, Kyoto, Japan, p. 86–97, 2000.

[HYP 00] Hyperion, "Analytical Processing : A Comparison of Multidimensional and SQL-based Approaches", *White Paper*, 2000.

[IKE 81] Ikeda H., Kobayashi Y., "Additional Facilities of a Concentional DBMS to Support Interactive Statistical Analysis", Wong H. K. T., Ed., *Proceedings of the First LBL Workshop on Statistical Database Management, Melno Park, California, USA, December 2-4, 1981*, Lawrence Berkeley Laboratory, p. 25-36, 1981.

[IMH 98] Imhoff C., "The Operational Data Store : Hammering Away", *DM Review*, http ://www.dmreview.com, 1998.

[INM 95] Inmon W., "The Operational Data Store", 1995, http ://www.evaltech.com/wpapers/ODS2.pdf.

[INM 96] Inmon W., *Building the Datawarehouse*, John Wiley & Sons, 1996.

[INM 00] Inmon W., "The Operational Data Store", *White Paper*, www.billinmon.com/library/whiteprs/earlywp/ttods.pdf, 2000.

[JAG 99] Jagadish H. V., Lakshmanan L., Srivastava D., "What can hierarchies do for data warehouses ?", *Proceedings of the International Conference on Very Large Databases (VLDB)*, Edinburgh, Scotland, p. 530–541, 1999.

[JAR 98] Jarke M., Lenzerini M., Vassiliou Y., Vassiliadis P., *Fundamentals of Data Warehouses*, Springer-Verlag, 1998.

[KAC 95] Kacprzyk J., Zadrozny S., "FQUERY for Access : fuzzy querying for a windows-based DBMS", Bosc P., Kacprzyk J., Eds., *Fuzzyness in Database Management Systems*, p. 415–433, Physica-Verlag (Springer Verlag), 1995.

[KAC 00a] KACPRZYK J., YAGER R., ZADROZNY S., "A fuzzy logic based approach to linguistic summaries of databases", *Int. Journal of Applied Mathematics and Computer Science*, vol. 10, p. 813–834, 2000.

[KAC 00b] KACPRZYK J., ZADROZNY S., "Computing with words : Towards a New Generation of Linguistic Querying and Summarization in Databases", SINČÁK P., VAŠČÁK J., KVASNIČKA V., MESIAR R., Eds., *Proc. of the Euro-International Symposium on Computational Intelligence (ISCI)*, vol. 54, Kosice, Slovaquie, Springer-Verlag, 2000.

[KAM 77] KAM J., ULLMAN J., "A Model of Statistical Databases and Their Security", *ACM Transactions on Database Systems (TODS)*, vol. 2, n° 1, p. 1-10, 1977.

[KAM 97] KAMBER M., HAN J., CHIANG J., "Metarule-Guided Mining of Multi-Dimensional Association Rules Using Data Cubes", *3rd Int. Conf. on Knowledge Discovery and Data Mining (KKD'97)*, Newport Beach, USA, p. 207-210, 1997.

[KAU 91] KAUFMANN A., GUPTA M., *Introduction to Fuzzy Arithmetic*, Van Nostrand Reinhohld, 1991.

[KEL 96] KELMAN A., Modèles flous pour l'agrégation de données et l'aide à la décision, PhD thesis, Université Paris VI, 1996.

[KIM 96] KIMBALL R., *The Datawarehouse Toolkit*, John Wiley & Sons, 1996.

[KLE 87] KLEIBER G., *Du côté de la référence verbale. Les phrases habituelles*, Éditions Peter Lang, 1987.

[KLE 90] KLEIBER G., *La sémantique du prototype*, Presses Universitaires de France, 1990.

[KLE 94a] KLEIBER G., *Catégorisation et hiérarchie : sur la pertinence linguistique des* termes *de base*, vol. 13, Hermes, 1994.

[KLE 94b] KLEIBER G., *Nominales. Essais de sémantique référentielle*, Collection Linguistique, Armand Collin, 1994.

[KLI 88] KLIR G., FOLGER T., *Fuzzy sets, uncertainty, and information*, Prentice Hall International, 1988.

[LAR 98] LARSEN H. L., ANDREASEN T., CHRISTIANSEN H., "Knowledge Discovery for Flexible Querying", ANDREASEN T., H.CHRISTIANSEN, LARSEN H., Eds., *FQAS*, n° 1495LNAI, Roskilde, Denmark, Springer, p. 227–235, 1998.

[LAR 99] LARSEN H. L., "An Approach to Flexible Information Access Systems using Soft Computing", *Proceedings of the Thirty-second Annual Hawaii International Conference on System Sciences*, Maui, Hawaii, 1999.

[LAU 00a] LAURENT A., BOUCHON-MEUNIER B., DOUCET A., GANCARSKI S., MARSALA C., "Fuzzy Data Mining from Multidimensional Databases", SINČÁK P., VAŠČÁK J., KVASNIČKA V., MESIAR R., Eds., *Proc. of the Euro-International Symposium on Computational Intelligence (ISCI)*, vol. 54, Kosice, Slovaquie, Springer-Verlag, p. 278–283, 2000.

[LAU 00b] LAURENT A., GANÇARSKI S., MARSALA C., "Coopération entre un système d'extraction de connaissances floues et un système de gestion de bases de données multidimensionnelles", *Rencontres Francophones sur la Logique Floue et ses Applications*, La Rochelle, France, Cepaduès editions, p. 325-332, 2000.

[LAU 01a] LAURENT A., "Bases de données multidimensionnelles floues", *Actes des Journées de bases de données avancées*, Agadir, Maroc, Hermes, p. 107-117, 2001.

[LAU 01b] LAURENT A., "De l'OLAP Mining au F-OLAP Mining", *Actes des Journées Francophones d'Extraction et Gestion des Connaissances, Revue Extraction des Connaissances et Apprentissage (ECA)*, vol. 1, n° 1–2, p. 189-200, 2001.

[LAU 01c] LAURENT A., "Generating Fuzzy Summaries from Fuzzy Multidimensional Databases", *Fourth International Symposium on Intelligent Data Analysis (IDA'01)*, vol. 2189 de *Lecture Notes in Computer Science*, Springer-Verlag, 2001.

[LAU 01d] LAURENT A., BOUCHON-MEUNIER B., DOUCET A., "Towards Fuzzy-OLAP Mining", *Proc. of the PKDD Workshop"Database Support for KDD"*, Fribourg, Allemagne, p. 51–62, 2001.

[LAU 02a] LAURENT A., BOUCHON-MEUNIER B., "Détection de cellules anormalement vides", *Actes des Journées Francophones d'Extraction et Gestion des Connaissances, Revue Extraction des Connaissances et Apprentissage (ECA)*, vol. 1, n° 4, 2002.

[LAU 02b] LAURENT A., BOUCHON-MEUNIER B., DOUCET A., "Flexible Unary Multidimensional Queries and their Combinations", *Proc. of the 9th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU'02)*, Annecy, France, p. 315–322, 2002.

[LAU 03] LAURENT A., "A new Approach for the Generation of Fuzzy Summaries based on Fuzzy Multidimensional Databases", *International Journal of Intelligent Data Analysis*, vol. 7, n° 2, IOS Press, 2003.

[LEF 80] LEFEBVRE J., *Introduction aux analyses statistiques multidimensionnelles*, Masson, 1980.

[LEH 98] LEHNER W., "Modelling Large Scale OLAP Scenarios", SCHEK H.-J., SALTOR F., RAMOS I., ALONSO G., Eds., *Proc. of the 6th International Conference on Extending Database Technology*, vol. 1377 de *Lecture Notes in Computer Science*, Valencia, Spain, Springer, p. 153-167, 1998.

[LEN 97] LENZ H., SHOSHANI A., "Summarizability in OLAP and Statistical Data Bases", IOANNIDIS Y., HANSEN D., Eds., *Proceedings of the Ninth International Conference on Scientific and Statistical Database Management (SSDBM)*, Olympia, Washington, USA, IEEE Computer Society, p. 132-143, 1997.

[LEV 02] LEVENE M., LOIZOU G., "Why is the star schema a good data warehouse design ?", *Information Systems*, vol. To appear, 2002.

[LI 96] LI C., WANG X., "A data model for supporting on-line analytical processing", *Proceedings Conference on Information and Knowledge Management (CIKM)*, Rockville, Maryland, USA, p. 81-88, 1996.

[LIM 98] LIM J., La fréquence et son expression en français, PhD thesis, Université Paris XIII, Villetaneuse, 1998.

[LIU 99] LIU H., LU H., FENG L., HUSSAIN F., "Efficient Search of Reliable Exceptions", *The Third Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD99)*, Beijing, China, p. 194-203, April 26-28 1999.

[LU 00] LU H., FENG L., HAN J., "Beyond Intra-Transaction Association Analysis : Mining Multi-Dimensional Inter-Transaction Association Rules", *ACM Transactions on Information Systems*, vol. 18, n° 4, p. 423–454, 2000.

[MAI 83] MAIER D., *The Theory of Relational Databases*, Computer Science Press, 1983.

[MAN 96] MANNILA H., "Data Mining : Machine Learning, Statistics, and Databases", SVENSSON P., FRENCH J., Eds., *Proceedings of the Statistical and Scientific Database Management*, Stockholm, Sweden, IEEE Computer Society, p. 2-9, 1996.

[MAR 98a] MARCEL P., Manipulation de Données Multidimensionnelles et Langages de Règles , PhD thesis, INSA Lyon, 1998.

[MAR 98b] MARSALA C., Apprentissage inductif en présence de données imprécises : Construction et utilisation d'arbres de décision flous, PhD thesis, LIP6, 1998.

[MAR 99a] MARCHISIO R., "HCR Technology – Hierarchical–Cartesian–Relational", *http ://151.36.192.138/HCRwp.pdf*, White Paper, 1999.

[MAR 99b] MARSALA C., BOUCHON-MEUNIER B., "An Adaptable System to Construct Fuzzy Decision Trees", *Proc. of the NAFIPS'99 (North American Fuzzy Information Processing Society)*, New York, USA, p. 223–227, June 1999.

[MED 94] MEDINA J., PONS O., VILA M., "GEFRED. A generalized model of fuzzy relational databases", *Information Science*, vol. 76, p. 87–109, 1994.

[MED 95] MEDINA J., VILA M., CUBERO J., PONS O., "Towards the implementation of a generalized fuzzy relational database model", *Fuzzy Sets and Systems*, vol. 75, p. 273–289, 1995.

[MIN 99] MINEAU G., LAHBOUB M., DRAPEAU G., DUSSAULT J., "A First Step Toward the Integration of a Decision-Tree Based Data Mining Technique with an OLAP System", *Proceedings of the 3rd Int. Conf. On the Practical Application of Knowledge Discovery and Data Mining (PADD-99)*, London, UK, p. 149–158, 1999.

[MOS 57] MOSTOWSKI A., "On a generalization of quantifiers", *Journal of Fund. Math.*, vol. 44, p. 12–36, 1957.

[MOU 95] MOUADDIB N., FOUCAUT O., "Handling of fuzzy information in object oriented environment", *Fuzziness in Database Management Systems*, Springer-Verlag, p. 139–165, 1995.

[MUN 99] MUNNEKE D., WAHLSTROM K., MOHANIA M., "Fragmentation of Multidimensional Databases", *Australian Database Conference*, p. 153-164, 1999.

[NET 99] NETZ A., "OLAP Services : Semiadditive Measures and Inventory - White Paper, Microsoft Corporation", 1999.

[NOL 99] NOLAN C., "Introduction to Multidimensional Expressions (MDX) - White Paper, Microsoft Corporation", 1999.

[OZS 83] OZSOYOGLU G., OZSOYOGLU Z., "Features of a System for Statistical Databases", HAMMOND R., MCCARTHY J., Eds., *Proceedings of the Second International Workshop on Statistical Database Management*, Lawrence Berkeley Laboratory, p. 9-18, 1983.

[PED 99a] PEDERSEN T. B., JENSEN C. S., DYRESON C. E., "Supporting Imprecision In Multidimensional Databases Using Granularities", *Proc. of the Eleventh International Conference on Scientific and Statistical Database Management (SSDBM99), Cleveland, Ohio, USA*, July 28-30 1999.

[PED 99b] PEDERSEN T., JENSEN C., "Multidimensional Data Modeling for Complex Data", *ICDE*, Sydney, Australia, p. 336-345, 1999.

[PED 00a] PEDERSEN T., Aspects of Data Modeling and Query Processing for Complex Multidimensional Data, PhD thesis, Aalborg University, 2000.

[PED 00b] Pedersen T., Jensen C. S., Dyreson C. E., "The TreeScape System : Reuse of Pre-Computed Aggregates over Irregular OLAP Hierarchies", *Proceedings of the Twenty-Sixth International Conference on Very Large Databases (VLDB00), Cairo, Egypt, September 10-14, 2000*, September 2000.

[PEN 95a] Pendse N., Creeth R., Database explosion, Report , The OLAP Report, 1995.

[PEN 95b] Pendse N., Creeth R., What is OLAP, An analysis of what the increasingly misused OLAP term, Report , The OLAP Report, 1995.

[PEN 01] Pendse N., "Multidimensional Data Structures", *White Paper*, the OLAP Report, 2001.

[PIC 65] Picard C., *Théorie des questionnaires*, Gauthier-Villars, 1965.

[PIL 95] PILOT, An introduction to OLAP, Multidimensional Terminology and Technology, Report , White Paper, available at http ://www.pilotsw.com/olap/olap.htm#dsgl, 1995.

[QUI 86] Quinlan J. R., "Induction of decision trees", *Machine learning, volume 1*, Kluwer Academic Publishers, p. 81–106, 1986.

[QUI 91] Quinlan J. R., "Unknown attribute values in induction", *Proc. of the Sixth International Machine Learning Workshop*, Morgan Kaufmann, p. 164–168, 1991.

[RAF 90] Rafanelli M., Shoshani A., "STORM : A Statistical Object Representation Model", *Statistical and Scientific Database Management*, p. 14-29, 1990.

[RAJ 88] Raju K., Majumdar A., "Fuzzy Functional Dependencies and Loss Less Join Decomposition on Fuzzy Relational Database Systems", *ACM Trans. on Database Systems*, vol. 13, n° 2, p. 129–166, 1988.

[RAS 00] Raschia G., Mouaddib N., "Evaluation de la qualité des partitions de concepts dans un processus de résumés de bases de données", *Rencontres francophones sur la logique floue et ses applications*, La Rochelle, France, Cépaduès Editions, p. 297-305, 2000.

[RIB 99] Ribeiro R., Moreira A., "Intelligent Query Model for Business Characteristics", Mastorakis N. E., Ed., *Computational Intelligence and Applications*, Greece, p. 277–283, 1999.

[RIF 96] Rifqi M., Mesures de comparaison, typicalité et classification d'objets flous : théorie et pratique, PhD thesis, Université Paris VI, 1996.

[RIF 98] Rifqi M., Monties S., "Fuzzy Prototypes for Fuzzy Data Mining", Pons O., Vila A., Kacprzyk J., Eds., *Knowledge Management in Fuzzy databases*, Physica Verlag, 1998.

[ROB 00] Roberts L., "Beyond Moore's Law : Internet growth trends", *IEEE Computer*, vol. 33, n° 1, p. 117–119, disponible sur internet dans une version intitulée *Internet growth trends* : http ://www.ziplink.net/~lroberts/IEEEGrowthTrends/IEEEComputer12-99.htm, 2000.

[ROT 96] Rotem D., Zhao J., "Extendible arrays for statistical databases and OLAP applications", Svensson P., French J., Eds., *8th International Conference on Scientific and Statistical Database Management*, Stockholm, Sweden, IEEE Computer Society, p. 108– 117, 1996.

[RUN 89] RUNDENSTEINER E., BIC L., "Aggregates in Possibilistic Databases", *Int. Conf. on Very Large Data Bases*, Amsterdam, Morgan Kaufmann, p. 287-295, 1989.

[RUN 92] RUNDENSTEINER E. A., BIC L., "Evaluating Aggregates in Possibilistic Relational Databases", *Data and Knowledge Engineering journal*, vol. 7, p. 239-267, 1992.

[SER a] DB2 OLAP Server, http ://www-3.ibm.com/software/data/db2/db2olap, White Paper.

[SER b] Microsoft SQL Server, http ://www.sql-server-performance.com.

[SER c] Oracle Express Server, http ://otn.oracle.com/products/exp_server/content.html.

[SHI 99] SHIEH S., LIN C., "Auditing user queries in dynamic statistical databases", *Information Sciences*, vol. 113, p. 131–146, 1999.

[SHO 82] SHOSHANI A., "Statistical Databases : Characteristics, Problems, and some Solutions", *Eigth International Conference on Very Large Data Bases, September 8-10, 1982, Mexico City, Mexico, Proceedings*, Morgan Kaufmann, p. 208-222, 1982.

[SHO 97] SHOSHANI A., "OLAP and Statistical Databases : Similarities and Differences", *Proceedings of the Sixteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, Tucson, Arizona, ACM Press, p. 185-196, 1997.

[SON 00] SONI S., KURTZ W., Optimizing Cube Performance Using Microsoft Analysis Services 2000, Report , UNISYS, 2000.

[SUZ 98] SUZUKI E., KODRATOFF Y., "Discovery of Surprising Exception Rules Based on Intensity of Implication", *Proceedings of the 2nd European Symposium on Principles of Data Mining and Knowledge Discovery (PKDD)*, vol. 1510 de *LNAI*, Nantes, France, p. 10–18, 1998.

[TAM 98] TAM Y., Data Cube : its implemantation and Application in OLAP Mining, Master thesis, Simon Fraser University, September 1998.

[THO 83] THOMAS J., HALL D., "ALDS Project : Motivation, Statistical Database Management Issues, Perspectives, and Directions", HAMMOND R., MCCARTHY J., Eds., *Proceedings of the Second International Workshop on Statistical Database Management*, Lawrence Berkeley Laboratory, p. 82-88, 1983.

[ULL 88] ULLMAN J., *Principles of Database and Knowledge-Base Systems*, vol. I., Computer Science Press, 1988.

[ULL 96] ULLMAN J., "Efficient Implementation of Data Cubes Via Materialized Views, A survey of the field", SIMOUDIS E., HAN J., FAYYAD U., Eds., *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD'96)*, Portland, Oregon, USA, AAAI Press, p. 386-388, 1996.

[U.M 96] U.M. FAYYAD G. PIATETSKI-SHAPIRO P. S., UTHURUSAMY R., *Knowledge Discovery and Data Mining*, AAAI Press, 1996.

[UMA 93] UMANO M., "FREEDOM-O : A Fuzzy Database system", DUBOIS D., PRADE H., YAGER R., Eds., *Readings in Fuzzy Sets for Intelligent Systems*, p. 667–675, Morgan Kaufmann Publisher, 1993.

[Vää 97] VÄÄNÄNEN J., "Generalized Quantifiers, an Introduction", *Proceedings of the 9th European Summer School in Logic, Language and Information (ESSLLI Workshop)*, vol. 9 de *Lecture Notes in Computer Science*, Aix-en-Provence, France, 1997.

[VAS 98] Vassiliadis P., "Modeling Databases, Cubes and Cube Operations", Rafanelli M., Jarke M., Eds., *Proceedings of the 10th International Conference on Scientific and Statistical Database Management (SSDBM)*, Capri, Italy, IEEE Computer Society, July 1998.

[VAS 99] Vassiliadis P., Sellis T., "A survey of logical Models for OLAP Databases", *SIGMOD Record*, vol. 28, n° 4, 1999.

[YAG 88] Yager R., "On ordered weighted averaging aggregation operators in multicriteria decision making", *IEEE Trans. on Systems, Man and Cibernetics*, vol. 18, n° 1, p. 183–190, 1988.

[YAG 93] Yager R. R., Larsen H., "Retrieving Information by Fuzzification of Queries", *Journal of Intelligent Information Systems (JIIS)*, vol. 2, n° 4, p. 421–441, 1993.

[YAZ 92] Yazici A., George R., Buckles B., "A survey of conceptual and logical models for uncertainty management", Zadeh, Kacprzyk, Eds., *Fuzzy Logic for the Management of Uncertainty*, Chapitre 31, p. 607–643, John Wisley, 1992.

[Y.L 98a] Y. Lin E. K. K., "An overview of fuzzy quantifiers. (I). Interpretations", *Fuzzy Sets and Systems*, vol. 95, n° 1, p. 1–21, 1998.

[Y.L 98b] Y. Lin E. K. K., "An overview of fuzzy quantifiers. (II). Reasoning and applications", *Fuzzy Sets and Systems*, vol. 95, n° 2, p. 135–146, 1998.

[ZAD 65] Zadeh L., "Fuzzy Sets", *Information and Control*, vol. 8, p. 338–353, 1965.

[ZAD 96] Zadeh L., "Fuzzy Logic = Computing with Words", *IEEE Transactions on Fuzzy Systems*, vol. 2, p. 103–111, 1996.

[ZAD 99] Zadeh L., "From Computing with Numbers to Computing with Words – From Manipulation of Measurements to Manipulation of Perceptions", *IEEE Transactions on Circuits and Systems*, vol. 45, p. 105–119, 1999.

[ZAI 98a] Zaiane O. R., Han J., Li Z., Hou J., "Mining Multimedia Data", *CASCON'98 : Meeting of Minds*, Toronto, Canada, p. 83–96, 1998.

[ZAI 98b] Zaiane O. R., Han J., Li Z. N., Chiang J. Y., Chee S., "MultiMediaMiner : A System Prototype for MultiMedia Data Mining", *ACM-SIGMOD Conf. on Management of Data, (system demo)*, 1998.

[ZAI 00] Zaiane O. R., Han J., Zhu H., "Mining Recurrent Items in Multimedia with Progressive Resolution Refinement", *Proceedings of the 16th International Conference on Data Engineering (ICDE'00)*, San Diego, California, USA, IEEE Computer Society, p. 461–470, 2000.

[ZHU 98] Zhu H., On-Line Analytical Mining of Association Rules, Master thesis, Simon Fraser University, December 1998.

# Annexe A

# Rappels sur la logique floue et le traitement des incertitudes

Les définitions des notions utilisées dans ce rapport sont rappelées ici. Les sous-ensembles flous ont été introduits par L. Zadeh [ZAD 65]. Pour une présentation plus détaillée, voir par exemple [BOU 95, BOU 94, KLI 88].

Dans le cadre de la théorie des sous-ensembles flous, un élément peut appartenir plus ou moins fortement à un ensemble. Étant donné un ensemble de référence $X$, un sous-ensemble flou $A$ de $X$ est alors défini par une fonction d'appartenance $f_A$ qui associe à chaque élément $x$ de $X$ le degré $f_A(x)$, compris entre 0 et 1, reflétant une gradualité dans son appartenance à A.

Une norme triangulaire (*t-norme*) est une fonction $\top : [0,1] \times [0,1] \rightarrow [0,1]$ commutative, associative, monotone, d'élément neutre 1.

Une conorme triangulaire (*t-conorme*) est une fonction $\perp : [0,1] \times [0,1] \rightarrow [0,1]$ commutative, associative, monotone, d'élément neutre 0.

Le tableau 1 propose quelques t-normes et t-conormes usuelles.

**Définition 23 (relation floue)** *Une relation floue $R$ entre deux ensembles de référence $X$ et $Y$ est définie comme un sous-ensemble flou de $X \times Y$. En particulier, si $X$ et $Y$ sont finis, elle peut être décrite par la matrice $M(R)$ des valeurs de sa fonction d'appartenance.*

**Définition 24 (réflexivité)** *Une relation floue $R$ définie sur $X^2$ est dite* réflexive *si pour tout $x \in X$, $f_R(x,x) = 1$. Elle est dite* anti-réflexive *si $\forall x \in X$, $f_R(x,x) = 0$.*

**Définition 25 (symétrie)** *Une relation floue $R$ définie sur $X^2$ est dite* symétrique *si $\forall (x,y) \in X \times X$ $f_R(x,y) = f_R(y,x)$. La relation est dite* antisymétrique *si $\forall (x,y) \in X \times X$ $f_R(x,y) > 0$ et $f_R(y,x) > 0 \Rightarrow x = y$.*

| | t-norme | t-conorme |
|---|---|---|
| Zadeh | $\min(x,y)$ | $\max(x,y)$ |
| probabiliste | $x.y$ | $x + y - xy$ |
| Lukasiewicz | $\max(x + y - 1, 0)$ | $\min(x + y, 1)$ |

TAB. 1 – Quelques t-normes et t-conormes usuelles

**Définition 26 (Composition de relations floues)** *La* composition *de deux relations floues $R_1$ sur $X \times Y$ et $R_2$ sur $Y \times Z$ définit une relation floue $R = R_1 \circ R_2$ sur $X \times Z$ de fonction d'appartenance :*

$\forall (x,z) \in X \times Z : f_R(x,z) = \sup_{y \in Y} *(f_{R_1}(x,y), f_{R_2}(y,z))$.

*L'opérateur * peut être par exemple une t-norme. Le plus classiquement utilisé est le min.*

**Définition 27 (transitivité)** *Une relation floue $R$ définie sur $X^2$ est dite* transitive *ssi $R \supseteq R \circ R$. En particulier, $R$ est* transitive max-min *ssi* $\forall (x,z) \in X \times X$
$f_R(x,z) \geq \sup_{y \in X} min(f_R(x,y), f_R(y,z))$.

**Définition 28 (relation floue d'ordre strict)** *Une relation floue d'ordre strict $R$ sur un ensemble $X$ est définie comme une relation :*

- *anti-réflexive,*
- *anti-symétrique,*
- *transitive.*

*Si $X$ est fini, elle peut être décrite par la matrice M(R) des valeurs de sa fonction d'appartenance.*

*En fait, il s'agit d'une relation floue où pour tout $x$ de $X$, $f_R(x,x) = 0$.*

**Définition 29 (Fermeture transitive)** *Soit $G$ le graphe associé à une relation floue d'ordre strict $R$ de fonction d'appartenance $f_R$. On appelle* fermeture transitive *la relation $R_T$ telle que $f_{R_T}(x,y) \neq 0$ si et seulement s'il existe un chemin dans $G$ d'origine $x$ et d'extrémité $y$.*

*Soit $R_T$ la fermeture transitive* max-min *de $R$ et $R \circ R$ la composition max-min de $R$ et $R$. On calcule $R_T$ suivant l'algorithme suivant :*

1. $R' = R$
2. *Tant Que $R'$ change*
   - $R = R'$
   - $R' = R \bigcup (R \circ R)$
3. *Stop.* $R_T = R'$

**Mesures de comparaison** [BOU 96]

Soit $\mathcal{F}(\Omega)$ l'ensemble des sous-ensembles flous de l'ensemble d'éléments $\Omega$ et $f_A$ la fonction d'appartenance du sous-ensemble flou A de $\mathcal{F}(\Omega)$.

**Définition 30 (Différence)** *Une opération sur $\mathcal{F}(\Omega)$ est appelée* différence, *notée $-$ si elle satisfait pour tout $A$ et pour tout $B$ dans $\mathcal{F}(\Omega)$ :*

1. *si $A \subseteq B$, alors $A - B = \emptyset$*
2. *$B - A$ est monotone en $B$ : si $B \subseteq B'$, alors $B - A \subseteq B' - A$*

**Définition 31 (Mesure d'ensemble flou)** *Une mesure d'ensemble flou $M$ est une fonction définie sur $\mathcal{F}(\Omega)$ et prenant ses valeurs dans $\mathbb{R}^+$ telle que, pour tout $A$ et pour tout $B$ dans $\mathcal{F}(\Omega)$ :*

*MI1 :* $M(\emptyset) = 0$

*MI2 : si $B \subseteq A$, alors $M(B) \leq M(A)$*

Étant donnée une mesure d'ensemble flou $M$, on définit une M-mesure de comparaison entre A et B comme une mesure qui prend en compte $M(A \cap B)$, $M(B-A)$ et $M(A-B)$ :

**Définition 32 (M-mesure de comparaison)** *Une* M-mesure *de comparaison sur* $\Omega$ *est une fonction* $S : \mathcal{F}(\Omega) \times \mathcal{F}(\Omega) \rightarrow [0,1]$ *telle que* $S(A,B) = F_S(M(A \cap B), M(B - A), M(A - B))$, *pour une fonction donnée* $F_S : \mathbb{R}^+ \times \mathbb{R}^+ \times \mathbb{R}^+ \rightarrow [0,1]$

**Définition 33 (Mesure de satisfiabilité)** *Une* M-mesure de satisfiabilité *sur* $\Omega$ *est une M-mesure de comparaison S satisfaisant :*

1. $F_S(u,v,w)$ *est non décroissante en u, non croissante en v et w.*
2. $F_S(0,v,w) = 0$ *quels que soient u et v.*
3. $F_S(u,v,w)$ *est indépendant de w*
4. $F_S(u,0,.) = 1$ *quel que soit* $u \neq 0$

On note $\mathcal{C}(A,B)$ la satisfiabilité de $A$ vis-à-vis de $B$ ($\mathcal{C}(A,B) = F_S(M(A \cap B), M(B - A), M(A - B))$). Exemples de mesures de satisfiabilité :

$$\mathcal{C}(A,B) = 1 - \sup_{f_A(x)=0} f_B(x) \tag{A.1}$$

ou encore

$$\mathcal{C}(A,B) = \inf_x \min(1 - f_B(x) + f_A(x), 1) \tag{A.2}$$

ou encore

$$\mathcal{C}(A,B) = \frac{M(A \cap B)}{M(A)} \tag{A.3}$$

$$\text{avec } M(A) = \begin{cases} \sum_{count}(A) = \sum_{x \in \Omega} f_A(x) & \text{si } \Omega \text{ est dénombrable} \\ \int_\Omega f_A(x)dx & \text{sinon} \end{cases}$$

# Annexe B

# Rappels sur les bases de données relationnelles

Le modèle relationnel est fondé sur la théorie des ensembles. Il a été proposé par Codd en 1970.

## Relation/table

On appelle *domaine* un ensemble de valeurs. Une *relation* est alors définie comme un sous-ensemble du produit cartésien d'une liste de domaines. On appelle *n-uplet* un élément d'une telle relation. Les relations sont représentées sous forme de tables, et on appelle *attribut* le nom donné à chaque colonne pour l'identifier de manière unique. L'arité d'une relation est donnée par le nombre d'attributs.

Un *schéma de relation* est défini par :

- un nom de relation,
- la liste des attributs associés à leurs domaines.

À un schéma peuvent donc correspondre plusieurs instances de relations.

## Base de données relationnelle

Une *base de données relationnelle* est formée par un ensemble de schémas de relations chacun associé à une instance.

Par convention, la *valeur nulle* est introduite pour représenter une information inconnue ou inapplicable.

## Manipulation des données

Il existe trois types de langage de manipulation des données :

- algèbre relationnelle,
- calcul relationnel de n-uplets,
- calcul relationnel de domaines.

Concernant l'algèbre relationnelle, l'ensemble des relations peut être muni d'opérations pour constituer une algèbre. Il existe cinq opérations de base, les autres opérations pouvant être exprimées à l'aide des premières :

– opérations unaires : projection, sélection
– opérations binaires : union, produit cartésien, différence
Les opérations dérivées sont : l'intersection, la $\theta$-jointure, et la division.

Le langage SQL (*(Structured Query Language*) est un langage de définition et de manipulation des données. Nous ne détaillons pas ici toutes les clauses possibles, mais donnons l'exemple de l'opérateur de regroupement *group by*, crucial dans le cadre des bases de données multidimensionnelles où il est alors associé à un opérateur d'agrégation.

Par exemple, on considère une table d'individus $IND$ suivante :

| NOM | AGE | VILLE |
|---|---|---|
| a | 18 | v1 |
| b | 16 | v1 |
| c | 17 | v2 |
| d | 18 | v2 |
| e | 16 | v1 |
| f | 18 | v2 |
| g | 17 | v1 |
| h | 17 | v2 |
| i | 18 | v1 |

La requête $SELECT * FROM\ IND\ GROUP\ BY\ AGE$ conduit au résultat suivant :

| NOM | AGE | VILLE |
|---|---|---|
| a | 18 | v1 |
| d | 18 | v2 |
| f | 18 | v2 |
| i | 18 | v1 |
| b | 16 | v1 |
| e | 16 | v1 |
| c | 17 | v2 |
| g | 17 | v1 |
| h | 17 | v2 |

En associant la requête à un opérateur d'agrégation ($SELECT\ AGE, VILLE, COUNT(*)$ $FROM\ IND\ GROUP\ BY\ AGE, VILLE$), on obtient la table suivante :

| AGE | VILLE | count |
|---|---|---|
| 18 | v1 | 2 |
| 18 | v2 | 2 |
| 16 | v1 | 2 |
| 17 | v2 | 2 |
| 17 | v1 | 1 |

Le cube construit à partir d'une telle requête est le suivant :

| | v1 | v2 |
|---|---|---|
| 18 | 2 | 2 |
| 17 | 1 | 2 |
| 16 | 2 | |

## Notions de clés

Une *clé primaire* d'une relation est un ensemble minimal d'attributs dont la connaissance des valeurs permet d'identifier de manière unique un n-uplet dans la relation consi-

dérée.

## Dépendances fonctionnelles

La *dépendance fonctionnelle* $X \rightarrow Y$ est valide sur la relation $R$ si et seulement si pour instance $r$ de $R$, on a :

$$\forall t_1 \in r, t_2 \in r \quad \pi_X(t_1) = \pi_X(t2) \Rightarrow \pi_Y(t_1) = \pi_Y(t2)$$

## Formes normales

On suppose une relation et un ensemble de dépendances fonctionnelles connues.

Une relation est en *première forme normale* si et seulement si tout attribut contient une valeur atomique.

Une relation est en *deuxième forme normale* si et seulement si :

- elle est en première forme normale ;
- tout attribut n'appartenant pas à une clé ne dépend pas que d'une partie de cette clé.

Une relation est en *troisième forme normale* si et seulement si :

- elle est en deuxième forme normale ;
- tout attribut n'appartenant pas à une clé ne dépend pas fonctionnellement d'un attribut non clé.

Une relation est en *forme normale de BOYCE-CODD (BCNF)* si et seulement si les seules dépendances fonctionnelles élémentaires[14] et non triviales[15] sont celles dont la partie gauche contient une clé.

## Schémas de données - Formes dénormalisées

La normalisation des bases permet de réduire le volume des données à stocker et les redondances (donc améliore les processus de mise à jour). Cependant, cette approche n'est pas adaptée aux bases décisionnelles qui requièrent alors de nombreuses et très coûteuses opérations de jointure. Des schémas dénormalisés ont donc été proposés pour les approches décisionnelles, les schémas dits *en étoile* et les schémas dits *en flocon*. Les données sont organisées autour de *tables de faits* décrivant les variables à analyser et de *tables satellites* décrivant les dimensions paramètres des analyses. La figure 1 illustre ces deux schémas.

[14] Soit $X$ et $Y$ deux ensembles d'attributs. On dit que la dépendance fonctionnelle $X \Rightarrow Y$ est élémentaire s'il n'existe pas $X' \subset X$ t.q. $X' \Rightarrow Y$ est valide.

[15] La partie droite n'apparaît pas dans la partie gauche.

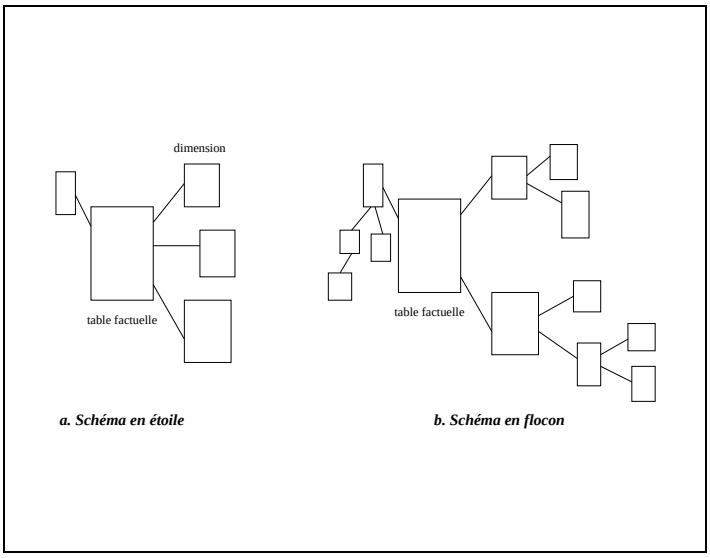


FIG. 1 – Schémas en étoile et en flocon

# Annexe C

# Démonstration de la transitivité de $\prec$

On rappelle la définition suivante pour les hiérarchies de dimensions :

**Définition**[Finesse de partitions floues] *On dit qu'une partition* $P_\alpha = \{F_1, \ldots, F_{L_\alpha}\}$ *est plus fine que la partition* $P_\beta = \{F'_1, \ldots, F'_{L_\beta}\}$ *dans la hiérarchie (et on note* $P_\alpha \prec P_\beta$*) si*

1. *Pour tout* $F' \in P_\beta$ *il existe un intervalle* $I \subset [1, L_\alpha]$ *tel que* $F' = \bigcup_{i \in I} F_i$,
2. $P_\alpha \neq P_\beta$.

On démontre que $\prec$ est une relation transitive.

Soit trois partitions $P_\alpha$, $P_\beta$ et $P_\gamma$, avec
$P_\alpha = \{F_i^\alpha\}_{i=1,\ldots,L_\alpha}$,
$P_\beta = \{F_i^\beta\}_{i=1,\ldots,L_\beta}$,
$P_\gamma = \{F_i^\gamma\}_{i=1,\ldots,L_\gamma}$,
où $P_\alpha \prec P_\beta$ et $P_\beta \prec P_\gamma$.

Par définition, pour tout $F \in P_\beta$, il existe un intervalle $I \in [1, L_\alpha]$ tel que $F = \bigcup_{i \in I} F_i^\alpha$
et pour tout $F \in P_\gamma$, il existe un intervalle $I \subset [1, L_\beta]$, tel que $F = \bigcup_{i \in I} F_i^\beta$
Alors pour tout $F \in P_\gamma$, il existe un intervalle $I \subset [1, L_\beta]$ et un ensemble d'intervalles $\{J_i \subset [1, L_\alpha]\}_{i \in I}$ tels que $F = \bigcup_{i \in I} \bigcup_{j \in J_i} F_j^\alpha$
On note $J = \bigcup_{i \in I} J_i$
$J$ est un intervalle ($J \subset [1, L_\alpha]$) puisque $I$ est un intervalle.
Donc pour tout $F \in P_\gamma$, il existe $J \subset [1, L_\alpha]$ tel que $F = \bigcup_{j \in J} F_j^\alpha$
Alors $P_\alpha \prec P_\gamma$

■

# Annexe D

# Démonstration de la transitivité de la généralisation pour le calcul des degrés

Dans le cas de généralisations successives, on a :

$$C \xrightarrow[\phi]{(dim,R,L_1,L_2)} C' \xrightarrow[\phi]{(dim,R,L_2,L_3)} C'' \tag{D.1}$$

$$C \xrightarrow[\phi]{(dim,R,L_1,L_3)} C''' \tag{D.2}$$

avec
- $dim$ la dimension à agréger,
- $R$ la relation d'ordre floue définissant la hiérarchie sur cette dimension (on note $f_R$ sa fonction et $f_{R_T}$ la fonction de sa fermeture transitive),
- $L_1$, $L_2$ et $L_3$ les différents niveaux (successifs) d'agrégation, et
- $\phi$ la fonction d'agrégation utilisée pour fusionner les valeurs des cellules.

Le détail de cette opération est donné page 88. Soit $c_n$ un élément du niveau $L_3$. En utilisant la première équation (D.1), on a :

$$d(c_n) = \max_{b_l \in L_2} \min(\max_{a_k \in L_1} \min(d(a_k), f_{R_T}(a_k, b_l)), f_{R_T}(b_l, c_n)) \tag{D.3}$$

En utilisant la seconde équation (D.2), on a :

$$d'(c_n) = \max_{a_k \in L_1} \min(d(a_k), \max_{b_l \in L_2} \min(f_{R_T}(a_k, b_l), f_{R_T}(b_l, c_n))) \tag{D.4}$$

On montre ci-dessous que $d'(c_n) = d(c_n)$.

On a :

$$\min(\max_{a_k \in L_1} \min(d(a_k), f_{R_T}(a_k, b_l)), f_{R_T}(b_l, c_n))$$

$$= \max_{a_k \in L_1} \min(d(a_k), f_{R_T}(a_k, b_l), f_{R_T}(b_l, c_n))$$

d'où

$$d(c_n) = \max_{a_k \in L_1} \max_{b_l \in L_2} \min(d(a_k), f_{R_T}(a_k, b_l), f_{R_T}(b_l, c_n))$$

De même, on a :

$$\min(d(a_k), \max_{b_l \in L_2} \min(f_{R_T}(a_k, b_l), f_{R_T}(b_l, c_n)))$$

$$= \max_{b_l \in L_2} \min(d(a_k), f_{R_T}(a_k, b_l), f_{R_T}(b_l, c_n))$$

d'où

$$d'(c_n) = \max_{a_k \in L_1} \max_{b_l \in L_2} \min(d(a_k), f_{R_T}(a_k, b_l), f_{R_T}(b_l, c_n))$$

soit $d'(c_n) = d(c_n)$

■

# Annexe E

# Démonstration de la propriété de coupure sur la génération des résumés flous

On étudie la proposition (1) de la page 131 rappelée ci-dessous.

**PROPOSITION**

Soit :

H.1 $Q$ un quantificateur de fonction d'appartenance $\mu_Q$ telle que $\forall x \in [0,1]$, $\mu_Q(x) > 0 \Rightarrow x > 0.5$,

H.2 $\{c_i\}_{i=1,\ldots,n}$ un ensemble de termes de résumé définis sur un univers $U$ décrits par leurs fonctions d'appartenance $\mu_i$ telles que $\forall u \in U \sum_{i=1}^{n} \mu_i(u) = 1$,

H.3 $s = 0.5$ le seuil de comptage. Une cellule à la position $\overrightarrow{x}$ n'est comptée que si son degré d'appartenance au cube $\mu(\overrightarrow{x})$ est strictement supérieur à $s$,

H.4 $\chi$ un critère et $|\chi|$ le nombre de cellules non vides $\overrightarrow{x}$ telles que $\mu(\overrightarrow{x}) > s$ après sélection par le critère $\chi$.

**S'il existe un résumé de la forme**

*Q des objets de la base qui sont* $\chi$ *sont* $c_i$ avec $\mu_Q\left(\frac{|\chi \cap c_i|}{|\chi|}\right) > 0$

**alors** il n'existe pas $c_j \neq c_i$ tel que *Q des objets qui sont* $\chi$ *sont* $c_j$ est vrai avec $\mu_Q\left(\frac{|\chi \cap c_j|}{|\chi|}\right) > 0$, c'est-à-dire que $\forall (j, j \neq i)\ \mu_Q\left(\frac{|\chi \cap c_j|}{|\chi|}\right) = 0$

On montre que cette proposition n'est vérifiée dans le cas général que si une condition, que nous introduisons en 1.2.2, est ajoutée.

## 1. Montrons que toute cellule d'un cube ne peut être comptée au plus que pour un seul critère

On considère le cube $C$ obtenu après sélection par $\chi$. On applique d'une part la sélection par le critère $c_i$ pour obtenir le cube $C_i$ et d'autre part la sélection par le critère $c_j$ pour

obtenir le cube $C_j$. Montrons que toute cellule du cube $C$ ne peut être à la fois retrouvée dans le cube $C_i$ et dans le cube $C_j$ avec un degré d'appartenance supérieur à 0.5 (par H.1).

Si le degré de vérité du résumé *Q des objets de la base qui sont* $\chi$ *sont* $c_i$ est supérieur à 0 alors $\frac{|\chi \cap c_i|}{|\chi|} > 0.5$

Soit $\mu(\overrightarrow{x})$ le degré d'appartenance de la cellule $\overrightarrow{x}$ contenant l'élément $(v(\overrightarrow{x}), d(\overrightarrow{x}))$. Après sélection par le critère $c_i$, le nouveau degré d'appartenance de la cellule au cube $C_i$, $\mu_{C_i}(\overrightarrow{x})$, est calculé comme suit :

$$\mu_{C_i}(\overrightarrow{x}) = \mathcal{T}(\mathcal{C}(\mu_i, v), d(\overrightarrow{x}), \mu(\overrightarrow{x}))$$

avec $\mathcal{C}(\mu_i, v) = \frac{M(c_i \cap v)}{M(v)}$ et $M(A) = \int_{u \in U} \mu_A(u) du$ pour toute valeur floue $A$ décrite par sa fonction d'appartenance $\mu_A$.

### 1.1. Valeurs non floues $v$

Pour chaque cellule $\overrightarrow{x}$, la valeur $v(\overrightarrow{x})$ est un nombre non flou. On note $\dot{v}(\overrightarrow{x})$ le singleton auquel est réduit le sous-ensemble flou. Le degré de satisfaction de la valeur $v$ au critère $c_i$ (resp. $c_j$) est $\mu_i(\dot{v}(\overrightarrow{x}))$ (resp. $\mu_j(\dot{v}(\overrightarrow{x}))$).

La cellule à la position $\overrightarrow{x}$ n'est comptée que si $\mu_{C_i}(\overrightarrow{x}) > 0.5$ (par H.3). On rappelle que tout t-norme $\top$ qui peut être utilisée pour construire $\mathcal{T}$ vérifie : $\forall(a,b)$, $\top(a,b) \leq min(a,b)$. Donc si on a $\mu_{C_i}(\overrightarrow{x}) > 0.5$ alors $\mu_i(\dot{v}(\overrightarrow{x})) > 0.5$. D'où $\mu_j(\dot{v}(\overrightarrow{x})) \leq 0.5$ (par H.2). La cellule ne peut donc pas être comptée après sélection par le critère $c_j$ (par H.1).

On a donc $|\chi \cap c_i| + |\chi \cap c_j| \leq |\chi|$

### 1.2. Valeurs floues $v$

La démonstration suit les mêmes étapes que précédemment. Pour chaque cellule $\overrightarrow{x}$, si la valeur $v(\overrightarrow{x})$ satisfait un critère avec un degré strictement supérieur à 0.5 alors elle ne peut satisfaire aucun autre critère avec un degré strictement supérieur à 0.5. Cette propriété est triviale dans le cas de partitions classiques (voir ci-dessous), mais devient beaucoup plus délicate dans le cas de partitions floues.

On suppose que $\mathcal{C}(c_i, v(\overrightarrow{x})) = \frac{\int_{u \in U} min(\mu_i(u), v(u)) du}{\int_{u \in U} v(u) du} > 0.5$

montrons que $\mathcal{C}(c_j, v(\overrightarrow{x})) = \frac{\int_{u \in U} min(\mu_j(u), v(u)) du}{\int_{u \in U} v(u) du} \leq 0.5$

#### 1.2.1. Cas de partitions classiques des termes de résumé (critères)

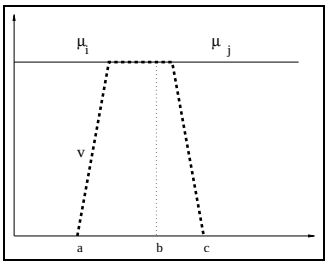


FIG. 1 – Partition non floue, valeur floue

On suit une interprétation géométrique (voir Fig. 1).

$\int_{u \in U} min(\mu_i(u), v(u))du$ est l'aire du polygone correspondant à l'intersection du critère $c_i$ et de la valeur $v$. $\int_{u \in U} v(u)du$ est l'aire du polygone correspondant à la valeur $v$ (il s'agit ici d'un trapèze mais toute autre forme est possible dans ce cas).

Si $c_j \cap v$ est vide alors $\mathcal{C}(c_j, v) = 0$

Sinon, on a :

$$\frac{\int_{u \in U} min(\mu_i(u), v(u))du}{\int_{u \in U} v(u)du} + \frac{\int_{u \in U} min(\mu_j(u), v(u))du}{\int_{u \in U} v(u)du} + \sum_{\substack{k=1 \\ k \neq i \\ k \neq j}}^{n} \frac{\int_{u \in U} \min(\mu_k(u), v(u))du}{\int_{u \in U} v(u)du} = 1$$

qui peut être interprété comme

$Aire(c_i \cap v) + Aire(c_j \cap v) + Aire(\overline{c_i} \cap \overline{c_j} \cap v) = Aire(v)$

D'où $\frac{\int_{u \in U} min(\mu_i(u), v(u))du}{\int_{u \in U} v(u)du} + \frac{\int_{u \in U} min(\mu_j(u), v(u))du}{\int_{u \in U} v(u)du} \leq 1$

Or $\frac{\int_{u \in U} min(\mu_i(u), v(u))du}{\int_{u \in U} v(u)du} > 0.5$

Donc $\frac{\int_{u \in U} min(\mu_j(u), v(u))du}{\int_{u \in U} v(u)du} \leq 0.5$

### 1.2.2. Cas de partition floue des termes de résumé

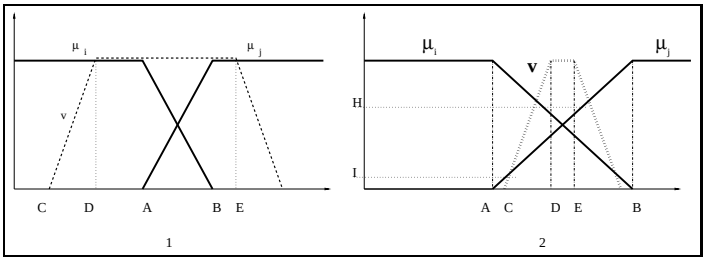


FIG. 2 – Partition floue, Valeur floue

On suppose que $v$ est une valeur floue trapézoïdale, symétrique, normalisée. On adopte une interprétation géométrique (voir Fig. 2).

Dans certains cas particuliers, la proposition est triviale.

Par exemple, on considère le cas illustré par la figure 2 (1er cas à gauche) où le noyau de la valeur $v$ recouvre les noyaux des deux critères $c_i$ et $c_j$ et où le segment défini par les extrémités du support de $v$ recouvre le segment $[AB]$.

On a $Aire(v \cap c_i) > \frac{Aire(v)}{2}$

c'est-à-dire $\frac{B-A}{2} + (A - D) + \frac{D-C}{2} > \frac{(A-D)+(B-A)+(E-B)+(D-C)}{2}$

donc $\frac{A-D}{2} > \frac{E-B}{2}$

$Aire(v \cap c_j) = \frac{B-A}{2} + (E - B) + \frac{D-C}{2}$

avec $(E - B) = \frac{E-B}{2} + \frac{E-B}{2}$ donc $E - B < \frac{E-B}{2} + \frac{A-D}{2}$

donc $Aire(v \cap c_j) < \frac{(A-D)}{2} + \frac{(B-A)}{2} + \frac{(E-B)}{2} + \frac{(D-C)}{2}$

donc $Aire(v \cap c_j) < \frac{Aire(v)}{2}$ d'où $\frac{Aire(v \cap c_j)}{Aire(v)} < 0.5$

Ainsi, une cellule du cube de départ comptée dans $C_i$ après sélection par le critère $c_i$ ne pourra pas être comptée dans $C_j$ par sélection de $c_j$.

Cependant, dans le cas général, la proposition n'est pas toujours vraie (voir par exemple la Fig. 2, 2ème cas à droite). Une condition supplémentaire est donc nécessaire pour démontrer le cas général.

On montre que la propriété étudiée est vraie ***si et seulement si*** la condition suivante est vérifiée :

**Condition.**

*Pour toute valeur $v$ d'un cube trapézoïdale normalisée symétrique, pour tous critères $c_i$ et $c_j$, $v$ est telle qu'en la centrant par translation sur l'intersection de $c_i$ et $c_j$ on a* $Aire(v \cap c_i) = Aire(v \cap c_j) \leq \frac{Aire(v)}{2}$.

**seulement si (condition nécessaire) -** On montre que $proposition(1) \Rightarrow condition$ par la contraposée $\neg$ condition $\Rightarrow \neg$ proposition(1).

Si la condition n'est pas vérifiée, alors en considérant deux critères $c_i$ et $c_j$ tels que $v$ soit centrée sur leur intersection, on a $Aire(v \cap c_i) > Aire(v)/2$ **et** $Aire(v \cap c_j) > Aire(v)/2$. Les deux critères sont vérifiés par $v$ avec un degré supérieur à 0.5, ce qui induit une contradiction due à des doubles comptages de cellules qui invalide la proposition (1).

**si (condition suffisante) -** On montre que la condition énoncée ci-dessus permet de valider la proposition (1) : $condition \Rightarrow proposition(1)$. Nous montrons qu'il n'existe pas deux critères $c_i$ et $c_j$ qui satisfont une valeur $v$ avec une degré supérieur à 0.5 quand la condition est satisfaite. Quand $v$ est centrée sur l'intersection de $c_i$ et $c_j$, aucun critère n'est satisfait avec un degré supérieur à 0.5. Pour que l'aire de $v \cap c_i$ soit supérieure à la moitié de l'aire de $v$, il faut donc que $v$ soit translatée du côté de $c_i$. Or l'aire de $v \cap c_j$ décroît quand on translate $v$ dans la direction de $c_i$, elle ne peut donc jamais devenir plus grande que la moitié de la valeur de $v$.

## 2. Conclusion

**Dans tous les cas**, si les conditions énoncées sont satisfaites, on montre que pour chaque cellule $\overrightarrow{x}$ du cube, si $\mu_{C_i}(\overrightarrow{x})$ est supérieur à 0.5 alors $\mathcal{C}(\mu_i, v(\overrightarrow{x})) > 0.5$. Ainsi, pour chaque critère $c_j$, $\mathcal{C}(\mu_j, v(\overrightarrow{x}))$ est inférieur ou égal à 0.5 donc $\mu_{C_j}(\overrightarrow{x}) \leq 0.5$. La cellule ne peut donc pas être comptabilisée pour les deux critères $c_i$ et $c_j$. D'où

$$|\chi \cap c_i| + |\chi \cap c_j| \leq |\chi|$$

**Donc** $\frac{|\chi \cap c_j|}{|\chi|} \leq 1 - \frac{|\chi \cap c_i|}{|\chi|}$
**Or** $\frac{|\chi \cap c_i|}{|\chi|} > 0.5$ **Alors** $\frac{|\chi \cap c_j|}{|\chi|} \leq 0.5$
**D'où** $\mu_Q\left(\frac{|\chi \cap c_j|}{|\chi|}\right) = 0$ ■

# Annexe F

# Questionnaires

Deux questionnaires ont été proposés à un ensemble de sujets afin d'étudier comment les analystes résument de façon linguistique les données numériques et d'étudier l'emploi des quantificateurs.

Les sujets ont été d'abord été soumis au premier questionnaire. Un deuxième passage a ensuite été réalisé en leur demandant de reprendre chaque tableau de données et de spécifier quels sont, selon eux, les quantificateurs qui peuvent être appliqués.

Pour faciliter la diffusion des questionnaires, ils ont été placés sur internet en utilisant des formulaires. Les sujets n'avaient qu'à remplir les questionnaires et les envoyer de manière simple (présence d'un bouton "ENVOYER").

Les sujets sont des personnes majeures, ayant toutes passé le baccalauréat.

# Questionnaire I

**QUESTIONNAIRE SUR LES RESUMES DE DONNEES**
**CE QUESTIONNAIRE PORTE SUR LES RESULTATS A UN EXAMEN (FICTIF).**

Dans le cadre de travaux de recherche sur la logique floue et les résumés de données, je voudrais étudier la façon dont nous associons spontanément une description linguistique à des données numériques. Votre aide me permettrait de mener à bien cette étude et de spécifier et comparer les différentes manières de résumer des données.

Pour participer à cette étude, il vous suffit de remplir le questionnaire ci-dessous et de l'envoyer (bouton "envoyer" à la fin du questionnaire). Je vous remercie d'avance pour votre aide. Pour tout renseignement, vous pouvez me contacter par mail.

**CONSIGNE**

Résumez chaque tableau en une phrase sans en reprendre les chiffres.

**REPARTITION DES INSCRITS**

*1. Répartition des inscrits selon le sexe*

| | |
|---|---|
| Garçons | 47,4% |
| Filles | 52,6% |

*2. Répartition des inscrits selon la nationalité*

| | |
|---|---|
| Français | 97% |
| Etrangers | 3% |

*3. Répartition des inscrits selon l'établissement d'origine*

| | |
|---|---|
| établissement public | 72% |
| établissement privé sous contrat | 19% |
| établissement privé hors contrat | 1% |
| inscrits à titre individuel | 4% |
| établissement de formation continue | 1,6% |
| apprentissage | 1,4% |
| CNED (enseignement à distance) | 1,4% |

*4. Choix des spécialités parmi les inscrits*

| | |
|---|---|
| pas de spécialité | 1% |
| latin ou grec | 0,7% |
| langue vivante | 14% |
| sciences économiques et sociales | 4% |
| mathématiques | 22% |
| physique-chimie | 7% |
| sciences de la vie et de la terre | 8% |
| biologie-écologie | 0,3% |
| pas de spécialité possible | 43% |

*5. Répartition des inscrits, examen scientifique*

| | |
|---|---|
| pas de spécialité | 4,1% |
| mathématiques | 34,3% |
| physique-chimie | 26,4% |
| sciences de la vie et de la terre | 34,2% |
| biologie-écologie | 1% |

*6. Répartition des inscrits, examen littéraire*

| | |
|---|---|
| latin | 4% |
| 2ème langue vivante | 0,6% |
| 3ème langue vivante | 29% |
| Langue renforcée | 31% |
| mathématiques | 25% |

*7. Répartition des inscrits selon le sexe*

| | |
|---|---|
| Garçons | 605 143 |
| Filles | 672 139 |

*8. PRESENCE par rapport aux inscrits*

| | |
|---|---|
| Présents | 98,3% |

*9. Obtention des mentions*

| Mention | taux d'obtention parmi les admis |
|---|---|
| sans mention | 52% |
| mention AB | 19% |
| mention B | 5% |
| mention TB | 0,7% |
| échec | 23,3% |

**REPARTITION DES ADMIS**

10.

| classe socio-professionnelle des parents | taux de réussite |
|---|---|
| Information, arts, spectacle | 85% |
| chômeur n'ayant jamais travaillé | 63% |

11.

| classe socio-professionnelle des parents | taux de réussite mention TB | taux de réussite sans mention |
|---|---|---|
| professeurs et assimilés | 3,7% | 55% |
| employé | 0,06% | 81% |

12.

| Sexe | Pourcentage parmi les admis |
|---|---|
| Garçons | 40,5% |
| Filles | 59,5% |

*13. Répartition des admis selon le type d'examen*

| | |
|---|---|
| scientifique | 41% |
| economique et social | 21% |
| littéraire | 20% |
| sciences et technologies industrielles | 4.60% |
| sciences et techniques de laboratoire | 0,93% |
| sciences et technologies tertiaire | 8.42% |
| sciences médico-sociales | 2,25% |
| hôtellerie | 0.36% |
| F11 | 0,08% |
| F12 | 0,30% |
| science et technologie aménagement | 0,76% |
| professionnel production | 0,01% |
| professionnel services | 0,02% |

*14. Obtention de l'examen*

| nationalité | Taux de réussite |
|---|---|
| étrangers | 67% |
| français | 77% |

*15. Obtention de l'examen*

| sexe | Taux de réussite |
|---|---|
| Féminin | 79,5% |
| Masculin | 74% |

*16. Obtention de la mention TB*

| Sexe | Obtention de la mention TB |
|---|---|
| Filles | 7,8% |
| Garçons | 7,6% |

*17. Obtention de la mention TB en fonction de l'examen passé*

| Examen | Proportion d'obtention mention TB |
|---|---|
| scientifique | 1,8% |
| Economique et Social | 0,4% |
| littéraire | 0,7% |

*18. Obtention de l'examen en fonction de l'année*

| ANNEE | TAUX DE REUSSITE |
|---|---|
| 1996 | 76% |
| 1997 | 78% |

## Questionnaire II

**CONSIGNE**

Pour chacun des tableaux de données, choisissez dans la liste déroulante le(s) mot(s) qui caractérise(nt) selon vous le mieux le tableau de données (utilisez la touche Ctrl pour les choix multiple names).

| |
|---|
| la plupart |
| peu de |
| environ la moitié |
| environ le tiers |
| environ le quart |
| toujours |
| jamais |
| souvent |
| tous |
| beaucoup |
| quelques |
| assez |
| généralement |
| habituellement |
| normalement |
| rare |
| nombreux |
| plus |
| chaque |
| une faible proportion de |
| un grand nombre de |
| presque tous |
| la majorité |

# Questionnaire III

**Reconstituez les tableaux de données numériques (pourcentages) à l'aide des résumés correspondants.**

**1. Répartition des inscrits selon le sexe**

Il y a un peu plus de filles que de garçons

| | |
|---|---|
| Garçons | % |
| Filles | % |
| **Total inscrits** | **100%** |

**2. Répartition des inscrits selon la nationalité**

Il y a très peu d'étrangers inscrits

| | |
|---|---|
| Français | % |
| Etrangers | % |
| **Total inscrits** | **100%** |

**3. Répartition des inscrits selon l'établissement d'origine**

Les inscrits proviennent majoritairement des établissements publics. Sinon, ils proviennent principalement d'un établissement privé sous contrat.

| | |
|---|---|
| établissement public | % |
| établissement privé sous contrat | % |
| établissement privé hors contrat | % |
| inscrits à titre individuel | % |
| établissement de formation continue | % |
| apprentissage | % |
| CNED (enseignement à distance) | % |
| **Total** | **100%** |

**4. Choix des spécialités parmi les inscrits**

Environ la moitié des inscrits n'ont pas de spécialité possible. La plupart des autres choisissent les maths ou une langue vivante.

| | |
|---|---|
| pas de spécialité | % |
| Latin ou grec | % |
| langue vivante | % |
| sciences économiques et sociales | % |
| mathématiques | % |
| physique-chimie | % |
| sciences de la vie et de la terre | % |
| biologie-écologie | % |
| pas de spécialité possible | % |
| **Total** | **100%** |

**5. Répartition des admis selon le sexe**

Les filles représentent une majorité d'admis

| | |
|---|---|
| Garçons | % |
| Filles | % |
| **Total** | **100%** |

**6.**

Le taux de réussite est plus élevé pour les français que pour les étrangers, tout en restant majoritaire.

| nationalité | Réussite | Echec | **TOTAL** |
|---|---|---|---|
| étrangers | % | % | **100%** |
| français | % | % | **100%** |

**7.**

Un quart des inscrits échouent. La moitié n'ont pas de mention. 1/5ème ont mention AB. Très peu ont une mention TB.

| Mention | taux d'obtention parmi les admis |
|---|---|
| Sans mention | % |
| mention AB | % |
| mention B | % |
| mention TB | % |
| échec | % |
| **TOTAL** | **100%** |

**8. Obtention de la mention TB en fonction du bac passé**

Le taux de mention TB est très faible en économique et social. Il est 3 fois plus important pour les bacs littéraires (un peu moins du double) et plus de 4 fois plus important pour les bacs scientifiques.

| Bac | mention TB | Autres mentions | **TOTAL** |
|---|---|---|---|
| bac scientifique | % | % | **100%** |
| bac Economique et Social | % | % | **100%** |
| bac littéraire | % | % | **100%** |

## Détail des questions (questionnaires I et II)

1. Valeurs très proches autour de la moitié
2. Valeurs très différentes l'une très grande l'autre très petite
3. Beaucoup de valeurs. Une très grande, 1 petite, toutes les autres très proches et très petites
4. Beaucoup de valeurs, mais pas de valeur très grande (max autour de la moitié). Les autres valeurs sont très petites mais pas aussi proches que précédemment
5. 2 valeurs très proches et une troisième pas très éloignée pas très élevées toutes les 3, et 2 autres proches et très basses
6. 3 valeurs proches autour du tiers et 2 autres très faibles (une pratiquement nulle)
7. Résultats en valeurs absolues. A comparer avec la question 1
8. Un chiffre très fort, le 2ème très faible
9. 2 proportions fortes (au-dessus de 50%) pas très proches mais quand même pas trop éloignées.
10. Sur les 2 lignes, même principe déséquilibre très fort entre colonnes. Avec en plus assez grosse différence entre les lignes avec croisement
11. 2 proportions assez proches autour de 50% mais l'une inférieure et l'autre supérieure de 10 points
12. Beaucoup de valeurs. La plupart très très petites et proches. 2 valeurs proches autour de 20% et une valeur seule autour de 40%
13. 2 valeurs proches assez moyennes
14. 2 valeurs proches fortes
15. 2 valeurs proches très faibles
16. 5 valeurs : 1 valeur autour de la moitié, 2 autour de 20% et les 2 autres très faibles
17. 3 valeurs très faibles et très proches (2 presque égales, la 3ème un tout petit peu plus élevée
18. 2 valeurs fortes très proches. A comparer à la question 14

# Annexe G

# Bases de test

Deux bases de données ont été utilisées pour valider le travail présenté dans cette thèse.

## Base *demo*

La base *demo* est la base fournie avec le système de gestion de bases de données multi-dimensionnelles Oracle Express. Elle est constituée d'un ensemble de cubes décrivant des résultat commerciaux. En particulier, le cube des *Ventes* décrit les résultats de ventes de produits en fonction du lieu géographique, du type de produit, et de la dimension temporelle. Chacune des trois dimensions du cube est munie d'une hiérarchie.

## Base *BAC*

La base *BAC* est une base fournie par le ministère de l'Éducation Nationale, de la Recherche et de la Technologie. Elle décrit les résultats au baccalauréat sur deux années consécutives et est structurée autour d'un schéma étoile.

Les cubes construits sont les suivants :

- Cube **CDeg** : ce cube décrit la proportion de candidats ayant obtenu une mention (AB-B ou TB, proportion par rapport au total des admis) par rapport aux dimensions
  - ANNEERENTREE (2 valeurs)
  - NATIONALITE (2 valeurs)
  - SERIEBAC (15 valeurs)
  - SEXE (2 valeurs)
  - SPECIALITE (12 valeurs)
  - STATUT (7 valeurs)
- **Cube PAdm** : ce cube décrit la proportion de candidats admis par rapport aux dimensions
  - AGE (44 valeurs)
  - NATIONALITE (2 valeurs)
  - PCS (39 valeurs)
  - ANNEERENTREE (2 valeurs)
  - SEXE (2 valeurs)
  - STATUT (7 valeurs)
  - SERIEBAC (15 valeurs)
  - SPECIALITE (12 valeurs)