Extraction De Motifs Séquentiels Dans Des Données Multidimensionelles

Marc Plantevit

Université Montpellier II
— Sciences et Techniques du Languedoc —

Thèse

pour obtenir le grade de
Docteur de l'Université Montpellier II

Discipline : Informatique
Spécialité Doctorale : *Informatique*
Ecole Doctorale : *Information, Structure, Systèmes*

présentée et soutenue publiquement par

Marc Plantevit

le 15 Juillet 2008

Extraction De Motifs Séquentiels Dans Des Données Multidimensionnelles

Jury

Dominique Laurent, Professeur, Université de Cergy Pontoise, ... Président
Jian Pei, Professeur, Simon Fraser University, Canada, ... Rapporteur
Karine Zeitouni, Maître de Conférence, Université de Versailles Saint Quentin-en-Yvelines, ... Rapportrice
Osmar Zaïane , Professeur, University of Alberta, Canada, ... Examinateur
Anne Laurent , Maître de Conférence, Université Montpellier II, ... Co-directrice de Thèse
Maguelonne Teisseire , Maître de Conférence, Université Montpellier II, ... Directrice de Thèse

Une thèse est le résultat d'un travail de longue haleine. C'est le fruit de multiples discussions au cours de diverses collaborations. L'organisation de ces idées qui constitue ma contribution modeste mais, je l'espère, originale, n'est pas possible sans un cadre de travail matériel et intellectuel favorable.

C'est pourquoi je tiens à remercier chaleureusement mes directrices de thèse Anne Laurent et[1] Maguelonne Teisseire qui m'ont donné d'excellentes conditions de travail. Qu'elles reçoivent toute ma gratitude pour m'avoir laissé si souvent le champ libre et m'avoir fait confiance dans mes choix et mes entêtements tout en m'apportant leurs rigueurs scientifiques. Mes premiers écrits se rappellent encore d'un certain stylo rouge qui devint ensuite noir. Pour cela et pour tout le reste, je leur suis extrêmement reconnaissant.

Je souhaite également remercier Jian Pei et Karine Zeitouni pour m'avoir fait l'immense honneur d'accepter d'être les rapporteurs de ce mémoire et pour le temps qu'ils ont consacrer à cette tâche. je remercie également Osmar Zaïane et Dominique Laurent pour avoir accepté de participer à l'évaluation de ce travail par leur présence lors de la soutenance de ce mémoire.

J'ai eu également plaisir à discuter avec d'autres chercheurs qui ne font pas partie des fonctions officielles de cette thèse. Merci donc à Jean Sallantin d'avoir participer au suivi de cette thèse, son expérience m'a été fort utile. Merci à Mathieu Roche, qui par son encore récent passé de doctorant, à su me guider dans la dernière ligne droite que représentent ces derniers mois de thèse. Je remercie également Pascal Poncelet qui a su me conseiller jusqu'au dernier et ultime moment. Je n'oublie pas les membres de l'équipe fouille de données du LIRMM.

Pendant, ces trois années, j'ai partagé mon bureau avec un compagnon de fortune (infortune ?). Merci donc à Chedy Raïssi (tac) pour les nombreuses discussions sur les tenants et les aboutissants de la fouille de données ainsi que les discussions parallèles et orthogonales. Trois ans de face-à-face, ça forge une amitié.

Je n'oublie pas mes prédécesseurs, Abdel, Simon, Lylia, Mehdi, Kristo, Jérôme, John, Clément, Céline et tous les autres qui ont su m'ouvrir la voie et dont les conseils furent très précieux dans ce long cheminement que représente une thèse. Une pensée également pour ce qui suivront, Alexandre, Flo, Nicolas, Antoine, Cécile, Lisa, Yoann, Julien, Hassan, etc.

Merci aux membres du Département SRC de l'IUT de Béziers pour leur accueil, leur soutien et leur confiance renouvelée lors de mes trois année de monitorat. Merci à Jérôme, Chrysta, Michel, Noëlle, Séverine, Ludovic, Chouki, Mathieu et Geneviève.

[1] Dans cette page de remerciements, le « *et* » est un opérateur commutatif.

Je remercie ma famille et mes amis qui m'ont soutenu patiemment pendant cette période.

Enfin, *last but not least*, merci à Laure d'avoir su m'accompagner et m'encourager depuis le début et ceci malgré mon manque de disponibilité.

A Laure

$\sim$

Soit A un succès dans la vie. Alors $A = x + y + z$, où $x =$ travailler, $y =$ s'amuser, $z =$ se taire.
Albert Einstein

Sommaire

Table des figures

Liste des Algorithmes

Introduction

Avec l'avènement des Technologies de l'Information et des Communications (TIC), les volumes de données brassés par les entreprises, les administrations et Internet sont devenus énormes. Les décideurs croulent désormais sous l'information à tel point qu'il est extrêmement difficile pour eux d'avoir une bonne vision des leurs données afin d'en tirer des bénéfices. Cette situation s'avère relativement paradoxale dans la mesure où les marchés sont de plus en plus concurrentiels et les entreprises se doivent d'être réactives. Cette réactivité passe par une grande expertise de ses données. Il est ainsi à la fois difficile et primordial pour un décideur d'analyser des giga-octets de transactions afin d'en dégager des informations utiles comme des tendances générales ou au contraire des exceptions.

La *fouille de données* (data mining) est apparue afin de permettre d'extraire des *connaissances* dans de grands volumes de données [CHY96]. Les connaissances extraites se traduisent par des informations utiles ou des *motifs* intéressants (non triviaux, implicites, présumés non connus et potentiellement utiles) qui permettent de tirer partie des données examinées afin d'aider l'utilisateur dans sa prise de décision. Selon le MIT (Massachusetts Institute of Technology), la fouille de données est l'une des technologies émergentes qui « changeront le monde » au XXI$^{\text{ème}}$ siècle.

Plus précisément, la fouille de données est une étape clé du *processus d'extraction de connaissances* dans les bases de données (ECD, KDD en anglais). Le processus d'ECD est illustré par la figure 1 [HK00]. Ce processus est divisé en trois étapes. La première, appelée *prétraitement*, prépare les données cibles pour pouvoir appliquer les techniques de fouilles de données. Cette étape inclut le nettoyage des données, l'intégration, la sélection et la transformation des données. L'étape principale du processus d'ECD est l'étape de fouille de données où différents algorithmes peuvent être appliqués afin de découvrir des connaissances. Après ce processus, l'étape de *post-traitement* évalue les résultats de l'extraction par rapport aux choix de l'utilisateur et aux connaissances du domaine. La connaissance peut être présentée à l'utilisateur si l'évaluation est satisfaisante, sinon il est nécessaire de relancer un ou tous les processus jusqu'à l'obtention de résultats satisfaisants.

Tout d'abord, il est nécessaire de nettoyer et d'intégrer les bases de données. Puisque les données peuvent être issues de différentes bases de données, il est possible d'avoir des inconsistances et des duplications. Il faut donc nettoyer les données sources en supprimant le bruit ou en faisant des compromis. Par exemple, supposons que nous avons deux bases de données, différents termes sont utilisés pour référer la même chose dans leur schéma. Nous pouvons également citer les données du monde réel qui sont

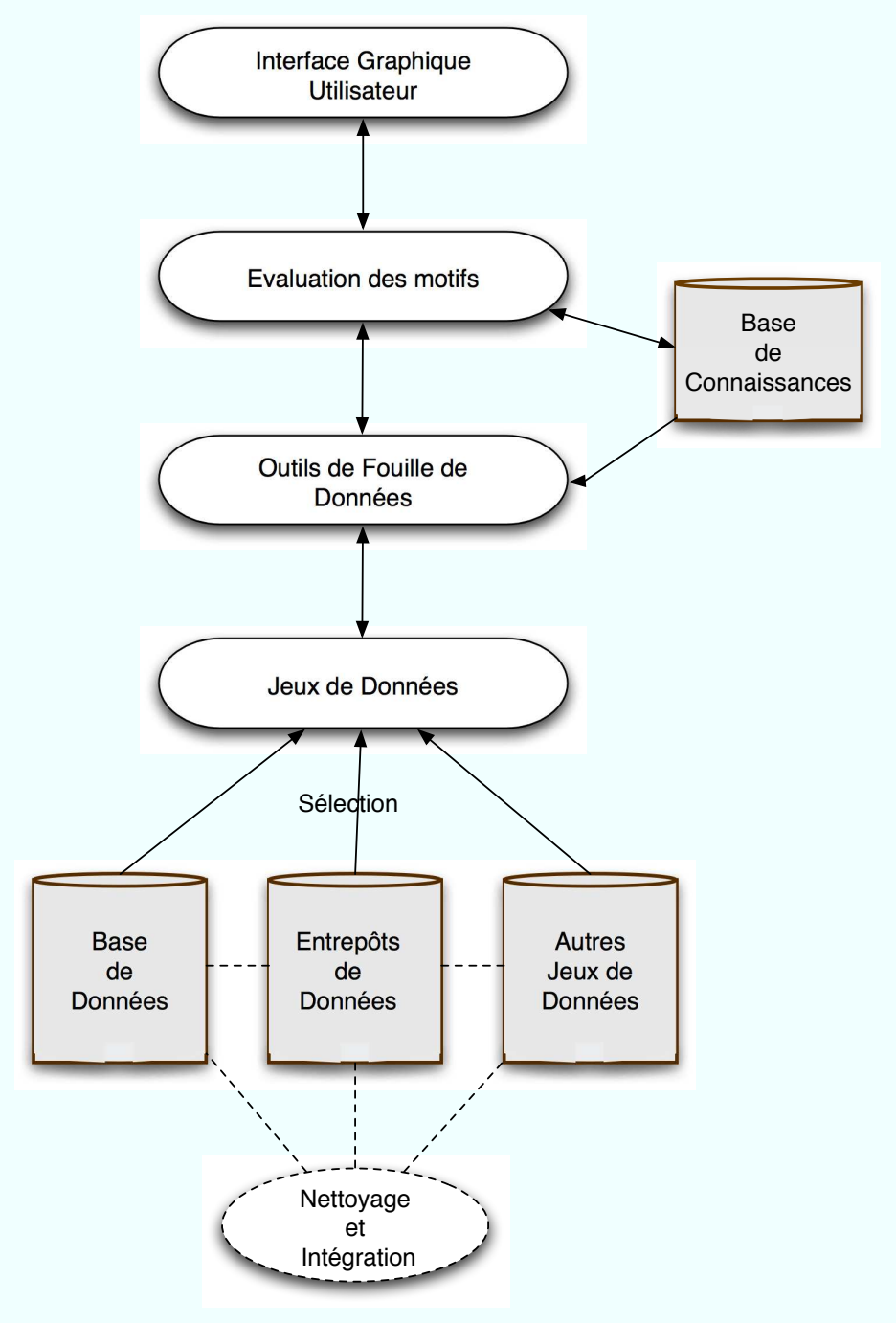


FIG. 1 – Le processus d'extraction de connaissances dans des bases de données (ECD)

souvent incomplètes et bruitées dues à leur enregistrement manuel. Les données intégrées sont ensuite stockées dans une base de données, un entrepôt de données ou d'autres types de collections (fichiers textes, XML, etc.).

La seconde étape vise à sélectionner les données pertinentes parmi les données intégrées et les transformer dans un format prêt à être analysé. Par exemple, supposons que nous voulons trouver les items qui apparaissent souvent ensemble dans les achats d'un supermarché. La base de données qui enregistre les ventes contient de nombreuses informations (identifiant du client, les items achetés, la date de la transaction, le prix, la quantité, etc.). Toutefois, nous n'avons besoin que des items achetés pour cette tâche spécifique. Après sélection des données pertinentes, la base de données qui va être fouillée est plus petite, ce qui assure, par conséquent, une extraction plus efficace.

Lorsque les données sont ordonnées selon une relation d'ordre, par exemple le *temps*, l'extraction de motifs séquentiels est bien adaptée [AS95]. En effet, les motifs séquentiels permettent d'établir des corrélations entre les événements au cours du temps. La plupart des données stockées par les entreprises, les administrations ou autres se prêtent à l'extraction de tels motifs. Cette technique permet, par exemple, d'extraire des motifs sur les habitudes de consommation des clients d'un magasin. En effet, un consommateur peut effectuer plusieurs achats à des dates différentes. On peut par exemple extraire des motifs de la forme *les clients achètent fréquemment un PC et des logiciels en même temps, puis reviennent pour acheter un appareil photo numérique et une carte mémoire, enfin ils achètent une imprimante et des livres sur la photographies.* L'extraction de motifs séquentiels peut s'appliquer dans de nombreux contextes autres que l'étude du panier de la ménagère. Ils peuvent être ainsi utilisés pour fouiller le comportements des internautes et permettre ainsi d'améliorer la structure du site concerné. Ils peuvent s'appliquer également en bioinformatique, en musique, sur du texte (l'orde sera par exemple celui des phrases dans le texte [JLT06]), etc.

Toutefois, les motifs séquentiels présentent des limites non-négligeables qui ne leur permettent pas d'exploiter pleinement les données qu'ils visent à décrire. En effet, les données sont souvent définies selon plusieurs dimensions ou attributs. Par exemple, les paquets réseaux sont définis avec des nombreuses options (adresse source, destination, TTL, flags, etc.), les cartes de fidélité des supermarchés permettent d'associer de nombreuses informations (âge, profession, loisirs, etc.) aux achats d'un consommateur en plus des informations contextuelles (date, heure, lieu, enseigne,etc.). Il est également courant que des relations hiérarchiques soient définies sur des dimensions. On peut, par exemple, facilement établir des relations hiérarchiques entre les produits d'un magasin (alimentations, boissons, boissons alcoolisées, sodas, etc.) ou sur des lieux géographiques (ville, département, région, etc.). Enfin, une autre spécificité doit être prise en compte : la présence de valeurs numériques. De telles valeurs peuvent par exemple définir la quantité de produits achetés par le consommateur lors d'une transaction ou la taille et le poids d'un individu. Dans les cubes de données, une ou plusieurs dimensions numériques appelées *mesures* permettent de représenter le résultat de l'agrégation d'un ensemble de faits.

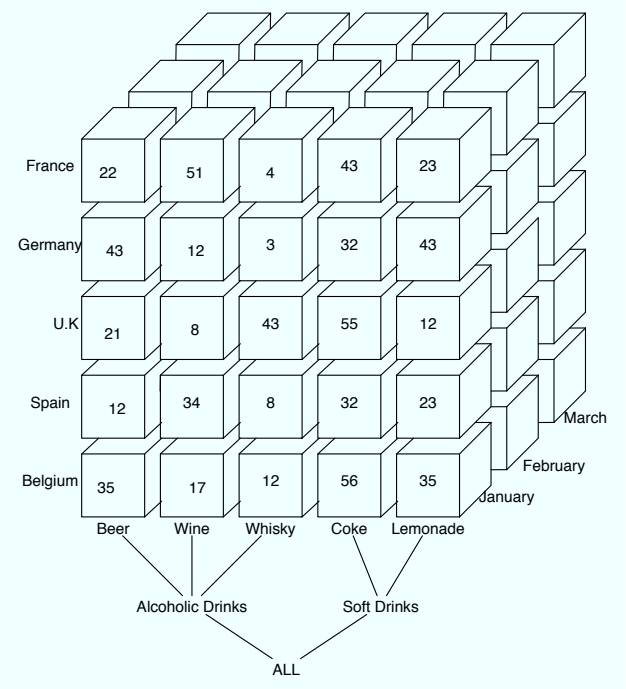


FIG. 2 – Un exemple de cube de données

Quasiment toutes les données réelles sont multidimensionnelles ou multi-attributs. Beaucoup sont également décrites à l'aide de hiérarchies ou disposent d'une ou plusieurs dimensions numériques. Les cubes de données qui offrent une représentation concise des tables de faits réunit par exemple toutes ces spécificités. La figure 2 représente un cube de données sur des ventes de boissons dans différents lieux. Outre la multidimensionnalité, les boissons sont décrites par l'intermédiaire d'une hiérarchie. Une dimension temporelle est présente. La *mesure* permet de représenter le résultat de l'agrégation des faits en fonction de différents paramètres. Par exemple, 35 *bières* ont été vendues en *Belgique* en *janvier*.

Il est nécessaire de prendre en compte toutes ces spécificités lors de l'extraction de motifs séquentiels multidimensionnels afin qu'ils offrent à l'utilisateurs une meilleure appréhension des données examinées. En effet, leur prise en compte peut permettre d'extraire des comportements généraux sur les données.Dans ce manuscrit, nous nous attachons à les prendre en compte. Nous détaillons plus précisément nos contributions et l'organisation de ce manuscrit dans les sections suivantes.

1 Contributions

Dans ce mémoire, nous étudions le problème de l'extraction des motifs séquentiels dans des contextes multidimensionnels. Les contributions associées à ce manuscrit sont décrites ci-dessous.

Une définition des motifs séquentiels multidimensionnels : Nous proposons une nouvelle définition des motifs séquentiels multidimensionnels. Le problème d'extraction de motifs séquentiels multidimensionnels associé devient alors une généralisation de l'extraction de motifs séquentiels

classiques et d'autres propositions. Ces motifs permettent de décrire des relations entre événements définis sur plusieurs dimensions ou attributs.

Des algorithmes d'extraction de motifs séquentiels multidimensionnels : L'espace de recherche associé à l'extraction de motifs séquentiels multidimensionnels est très important. Nous proposons différents algorithmes pour extraire efficacement des motifs séquentiels multidimensionnels. L'algorithme M^2SP cible une certaine partie de l'espace de recherche en recherchant les motifs séquentiels multidimensionnels à partir des items les plus spécifiques. Les algorithmes $CMSP\_Cand$ et $CMSP\_Free$ proposent d'extraire des motifs séquentiels multidimensionnels *clos*. L'utilisation d'une représentation *condensée* permet ainsi d'introduire de nouvelles propriétés d'élagage de l'espace de recherche et d'extraire un ensemble non redondant de motifs séquentiels multidimensionnels.

La prise en compte des hiérarchies dans l'extraction de motifs séquentiels multidimensionnels : Les données sont souvent définies selon plusieurs niveaux de hiérarchies. Les hiérarchies permettent de définir des relations entre les valeurs de chaque dimension. Nous proposons de prendre en compte les hiérarchies dans l'extraction de motifs séquentiels multidimensionnels. Deux algorithmes sont définis et proposent une gestion différente des hiérarchies dans l'extraction des motifs séquentiels multidimensionnels *h-généralisés*.

La prise en compte de valeurs numériques : Dans certains contextes, une dimension numérique peut apparaître. Cette dimension peut représenter le résultat de l'agrégation de faits définis sur plusieurs dimensions. Nous proposons plusieurs façons de prendre en compte de telles dimensions numériques.

Des méthodes d'extraction de comportements atypiques : L'extraction de motifs séquentiels multidimensionnels permettent d'extraire les comportements généraux ainsi que des tendances sur les données examinées. De telles connaissances peuvent s'avérer utiles dans de nombreux cas. Toutefois, dans certains contextes, il est nécessaire de découvrir également les comportements atypiques. Nous considérons qu'un comportement atypique peut être soit une séquence de données atypique (outlier) soit une connaissance qui se démarque des autres (connaissances inattendues). Nous proposons des algorithmes pour détecter chacune de ces interprétations.

Ces différentes propositions sont validées par des expérimentations sur des jeux de données synthétiques et réels. Les jeux de données réels, issus d'EDF, sont décrits dans l'annexe A. Ces jeux de données sont fournis par Françoise Guisnel, Sabine Goutier et Marie-Luce Picard, d'EDF R&D, dans le cadre d'une collaboration scientifique visant à découvrir des comportements atypiques dans des cubes de données.

2 Organisation du manuscrit

Ce manuscrit s'organise de la façon suivante.

Dans le chapitre 1, nous présentons un panorama des travaux de la littérature connexes à notre problématique.

Dans la partie I, nous présentons une nouvelle définition des motifs séquentiels multidimensionnels dans le chapitre 1. Le chapitre 2 introduit l'algorithme M^2SP alors que le chapitre 3 propose d'extraire des motifs séquentiels multidimensionnels clos à l'aide des algorithmes $CMSP\_Cand$ et $CMSP\_Free$.

La partie II traite la prise en compte des spécificités inhérentes aux données multidimensionnelles (hiérarchies, valeurs numériques). Le chapitre 1 définit les motifs séquentiels multidimensionnels *h-généralisés* qui sont définis sur plusieurs niveaux de hiérarchies. L'algorithme M^3SP permet d'extraire de tels motifs à partir des items *h-généralisés* les plus spécifiques. Le chapitre 2 propose un autre point de vue sur la prise en compte des hiérarchies dans l'extraction de motifs séquentiels multidimensionnels en proposant l'extraction de motifs séquentiels multidimensionnels *convergents* ou *divergents*. Enfin, nous proposons de prendre en compte les valeurs numériques dans le chapitre 3 de trois façons différentes.

La partie III s'intéresse à la découverte de comportements temporels atypiques dans des données multidimensionnelles. Notons que le temps est une excellente illustration d'une relation d'ordre. Par abus de langage, nous utilisons le terme temporel à la place du terme plus général séquentiel. Le chapitre 1 propose une méthode de navigation dans des cubes de donnés basée la découverte de séquence outlier. Dans le chapitre 2, nous définissons les règles séquentielles multidimensionnelles *inattendues*. De telles règles sont cachées par des règles ayant un support important. Elles identifient des comportements qui se démarquent des autres. Nous proposons un algorithme permettant leur extraction.

Enfin, nous finissons par un bilan où nous discutons notamment des perspectives associées aux différentes propositions de ce manuscrit.

Chapitre 1

Etat de l'Art sur l'Extraction de motifs

Dans ce chapitre, nous présentons les approches de la littérature proches de notre problématique. Nous définissons d'abord des critères d'analyse. Nous présentons ensuite ces approches. Enfin, nous discutons ces approches par rapport à notre problématique.

1 Introduction et critères d'évaluation

Les motifs fréquents sont des connaissances extraites sur des données. Leur but est de fournir à l'utilisateur des informations non triviales, implicites, présumées non connues et potentiellement utiles. Ils offrent ainsi à l'utilisateur une meilleure appréhension des données. Les motifs fréquents sont des itemsets, des sous-séquences, des sous arbres ou des sous-graphes apparaissant dans un jeu de données et vérifiant un seuil de support minimum fixé par l'utilisateur. Par exemple, un ensemble d'items, tels que *lait* et *pain*, qui apparaissent fréquemment ensembles dans des transactions d'un jeu de données est un *itemset*. Une sous-séquence, telle que *acheter d'abord un PC puis un appareil photo numérique et ensuite une carte mémoire*, qui apparaît fréquemment dans une base de transactions est un *motif séquentiel (fréquent)*.

La découverte de motifs joue un rôle clé dans la recherche d'associations, de corrélations et d'autres relations entre les données. De plus, la découverte de motifs peut aider l'indexation des données, la classification, le clustering et d'autres techniques de fouille de données. L'extraction de motifs fréquents est ainsi devenue une tâche importante de la fouille de données et un thème très étudié dans la communauté.

L'extraction de motifs fréquents a d'abord été proposée par [AIS93] (algorithme *Apriori*) avec les règles d'association pour l'étude du « panier de la ménagère ». [AIS93] analyse les habitudes des consommateurs en découvrant des corrélations entre les articles présents dans les caddies. Par exemple, si un consommateur achète du lait alors il achète également des céréales ou du pain. Des telles informations peut

permettre d'augmenter les ventes en étant les commerçants a proposer des promotions adaptées et organiser les rayons du supermarché. Rapidement, la prise en compte d'une relation d'ordre (le temps) s'est avérée primordiale. [AS95] propose ainsi les motifs séquentiels qui permettent d'établir des corrélations entre des événements suivant leur chronologie d'apparition.

Depuis ces deux articles fondateurs, des centaines de travaux ont été publiés, proposant de nombreuses extensions et applications. Dans ce chapitre, nous effectuons un panorama des différents travaux selon certains critères que nous exposons et justifions.

La prise en compte d'une relation d'ordre : La prise en compte d'une relation d'ordre (e.g. le temps) permet d'établir des corrélations entre des événements au cours de cette relation. Dans cette thèse, nous nous attachons à prendre en compte ce facteur.

La multidimensionnalité : De nombreux travaux proposent d'établir des corrélations entre des valeurs d'un unique attribut (les produits d'un supermarché,etc.). Combiner plusieurs dimensions ou attributs peut pourtant permettre d'obtenir des motifs décrivant mieux les données, offrant ainsi une meilleure appréhension des données sources.

Les hiérarchies : Les hiérarchies permettent de représenter les données à différents niveaux de granularité. Les données réelles sont souvent décrites par des relations hiérarchiques. De plus, un autre problème connexe à l'extraction de motifs est le choix de la bonne valeur du seuil de support minimum. Si le seuil de support est trop élevé, peu de motifs sont extraits et les motifs extraits sont souvent trop généraux et déjà connus pour être utiles. Si le seuil de support minimum est trop faible, alors un nombre très (trop) important de motifs est extrait, rendant sa gestion très difficile par l'utilisateur. L'utilisation des hiérarchies peut permettre de résoudre ce dilemme.

Les valeurs numériques : L'extraction de motifs est généralement associée à des attributs symboliques. Toutefois, il arrive que des attributs numériques soient bien adaptés à l'extraction de motifs. Par exemple, les quantités peuvent permettre d'affiner les motifs en décrivant dans quelles quantités les items du motif sont décrits.

Nous nous attachons ci-dessous à décrire et critiquer les approches d'extraction de motifs en fonction de ces critères. Comme la prise en compte d'une relation d'ordre (temps) est primordiale dans notre problématique, nous ne décrirons pas les nombreux travaux effectués sur l'extraction d'itemsets, le lecteur intéressé par un panorama des travaux d'extraction d'itemsets peut se référer à [Goe03].

2 Règles d'association dans des bases de données multidimensionnelles

Dans [KHC97], les auteurs sont les premiers à aborder le problème de l'extraction de règles d'association dans les bases de données multidimensionnelles. Ils introduisent une approche de fouille guidée de règles d'association. Cette proposition s'appuie sur un modèle général (méta-règle), qui définit le contenu

des règles d'association recherchées. Cette méta-règle définit une conjonction de prédicats dans le prémisse et la conséquence des règles recherchées. Chaque prédicat pose une condition sur une dimension. La méta-règle va ainsi permettre de guider le processus d'extraction de règles *inter dimensionnelles*. C'est à dire que des corrélations entre des positions (valeur d'une dimension) sont extraire.

Cette approche ne permet pas d'extraire des règles d'association définies sur plusieurs niveaux de hiérarchies. De plus, elle ne permet que d'extraire des corrélations entre différentes positions d'une cellule. Des corrélations entre différentes valeurs d'une même dimension, comme les règles d'association classiques, ne peuvent pas être extraites. Par exemple, pour deux dimensions *Age* et *Produit*, cette approche permet d'extraire des corrélations entre une valeur de *Age* et une valeur de *Produit*, elle ne permet pas d'extraire des corrélations entre différentes valeurs de *Produit*. La prise en compte du temps n'est pas abordée par cette proposition. Cette approche se situe dans des bases de données multidimensionnelles et s'appuie sur des opérateurs d'agrégation pour calculer le support et la confiance des règles, nous pouvons donc considérer qu'elle traite les valeurs numériques.

[Zhu98] propose d'extraire des règles *inter dimensionnelles, intra dimensionnelles* et des règles *hybrides*. Le cube de données est transformé sous forme tabulaire. Les motifs fréquents sont alors extraits à l'aide l'algorithme *Apriori* et les règles d'associations sont ensuite générées. Cependant, cette approche ne permet ni de prendre en compte le temps, ni des hiérarchies.

Dans [IKA02], les auteurs proposent une généralisation des règles d'association, nommée *Cubegrades*, qui permet de découvrir les changements de la mesure à la suite d'une modification des valeurs des dimensions d'un cube. Cette approche permet de chercher des conséquences en fonction de causes (*et si ...*) en calculant le différentiel de mesure des agrégats d'un cube de données suite à des opérations de spécialisation, de généralisation ou de modification (instanciation différente d'une dimension). Les auteurs proposent ainsi d'utiliser plusieurs opérateurs d'agrégation autres que le comptage habituellement utilisé (somme, etc.). Notons toutefois, que la notion de motif n'est plus la même si d'autres opérateur sont utilisé (la notion de fréquence disparaît).

Cette approche permet de prendre en compte les hiérarchies dans un contexte multidimensionnel. La prise en compte du temps n'est pas abordée.

Basé sur les mêmes motivations, [DHL+01] propose les *multidimensional constraint gradients* afin de détecter les changements significatifs des mesures lors d'une généralisation, d'une spécialisation ou d'une instanciation.

Dans [NJ03], les auteurs introduisent la notion de *règle d'association étendue* qui permet de contextualiser une règle d'association classique. Par exemple, la règle d'association classique sac de couchage $\Rightarrow$ tente est contextualisée avec les description période $=$ été et région $=$ Nord.

Cette proposition ne permet pas d'extraire des corrélations entre les événements au cours du temps.

Dans [TT04], les auteurs proposent d'extraire des règles d'association inter dimensionnelles. Ils effectuent un prétraitement des données multidimensionnelles afin d'extraire les règles à l'aide de l'algorithme

Apriori. Cette approche permet de prendre en compte les hiérarchies, mais ne peut pas prendre en compte le temps.

Enfin, [MRBM06] propose une nouvelle définition du support et de la confiance des règles d'association inter dimensionnelles basées sur la *somme* comme opérateur d'agrégation. Ces nouvelles définitions permet d'enrichir la sémantique des règles et de les rendre plus adaptées au contexte OLAP. Toutefois, cette proposition ne permettent ni de gérer les hiérarchies ni d'extraire des corrélations intra dimensionnelles ou temporelles.

3 Motifs colossaux

Le développement de la bioinformatique a engendré de nouveaux jeux de données ayant des caractéristiques originales. Par exemple, les *microarrays* décrivent les niveaux d'expression des gènes. La principale particularité des ces données est le nombre extrêmement important de dimensions (i.e., de l'ordre de 10000 à 100000 colonnes). Si nous prenons chaque échantillon comme une transaction et l'expression de chaque gène comme un item, la table est extrêmement large par rapport aux autres tables de transactions (supermarché, etc.). De tels jeux de données représentent de nouveaux challenges pour les algorithmes d'extraction de motifs puisqu'ils ont un nombre exponentiel de combinaisons entre items en fonction du nombre de transactions.

Dans [PCT+03], les auteurs proposent CARPENTER, une méthode pour extraire des motifs clos dans des jeux de données biologiques contenant un très grand nombre de dimensions. Cette approche intègre à la fois les avantages de la représentation des données sous forme verticale et le paradigme *pattern growth*. En convertissant les données sous forme verticale (item : ensemble des identifiants des transactions), un arbre *FP-Tree* [HPY00] peut être construit et énumérer toutes les transactions efficacement. CARPENTER parcourt ensuite l'arbre d'énumération des transactions en profondeur et extrait les motifs clos en examinant les items associés à chaque nœud. Définie dans [PTCX04], la méthode $COBBLER$ intègre l'énumération des itemsets à l'énumération des transactions.

Dans [LHXS06], les auteurs proposent *TD-Close* pour extraire l'ensemble complet des itemsets fréquents clos dans des données contenant beaucoup de dimensions. Cette approche se base sur une énumération des transactions en partant de l'ensemble maximal de transactions (l'ensemble contenant toutes les transactions). On appelle un tel parcours *top-down* car on part de l'élément le plus spécifique du treillis associé (haut) et on parcourt l'espace de recherche du plus spécifique vers le plus général (bas). Ce parcours permet d'utiliser le seuil de support minimum pour élaguer l'espace de recherche. De plus, une propriété de fermeture est utilisée afin d'éviter des passes inutiles sur le jeu de données.

Dans [ZYH+07], les auteurs montrent les limites des approches précédentes. Ces approches ne permettent pas encore d'extraire des *motifs colossaux* puisqu'elles s'appuient sur la génération d'un nombre potentiel exponentiel de motifs de *petite taille*, ce qui rend très difficile l'extraction de motifs colossaux avec de telles méthodes. [ZYH+07] propose une nouvelle approche, nommée *Pattern-Fusion*, pour ex-

traire efficacement les motifs colossaux par approximation. Ainsi, un motif colossal peut être découvert en une seule passe en fusionnant des *fragments* de motifs.

Les approches précédentes s'avèrent remarquables pour traiter des jeux de données où le nombre de dimensions est très élevé. Toutefois, ces approches ne permettent pas la prise en compte d'une relation d'ordre. De plus, même si les données sont décrites selon un très grand nombre de dimensions, la cardinalité de ces dimensions est très faible. Du fait de la spécificité des données, aucune approche ne propose de gérer des hiérarchies ou des valeurs numériques.

4 Motifs séquentiels

Une base de données séquentielles contient un ensemble ordonné d'éléments ou d'événements, enregistrés avec ou sans valeur concrète du temps. On retrouve de telles séquences dans de nombreuses applications comme les séquences d'achats des consommateurs, les séquences biologiques. L'extraction de motifs séquentiels a été introduite par [AS95] et est devenue depuis un domaine très actif de la fouille de données. Nous introduisons d'abord les concepts préliminaires relatifs aux motifs séquentiels.

Soit $I = \{i_1, i_2, \ldots, i_k\}$ l'ensemble de tous les items. Un sous-ensemble de I est appelé un *itemset*. Une *séquence* $\alpha = \langle a_1, a_2, \ldots, a_n \rangle$ est une liste ordonné d'itemset ($a_i \subseteq I$). Chaque itemset d'une séquence représente un ensemble d'événements qui apparaissent à la même estampille temporelle. Les différents itemsets d'une séquence sont associées à des estampilles temporelles différentes. Par exemple, un consommateur peut acheter plusieurs produits lors d'un passage dans le magasin et revenir plusieurs fois faire des achats. Il peut ainsi acheter un PC et des logiciels puis revenir acheter un appareil photo numérique avec une carte mémoire puis enfin acheter une imprimante et des livres sur la photographie.

Une séquence $\alpha = \langle a_1, a_2, \ldots, a_m \rangle$ est une *sous-séquence* de $\beta = \langle b_1, b_2, \ldots, b_n \rangle$ (noté $\alpha \preceq \beta$) si et seulement si $\exists i_1, i_2, \ldots, i_m$ tels que $1 \leq i_1 < i_2 < \ldots < i_m \leq n$ et $a_1 \subseteq b_{i_1}, \ldots, a_m \subseteq b_{i_m}$. On dit également que β est une super séquence de α ou que β contient α.

Etant donné un ensemble de séquences $D = \{s_1, s_2, \ldots, s_n\}$, le *support* d'une séquence α correspond au nombre de séquences de D qui contiennent α. Si le support d'une séquence α satisfait un seuil de support minimum $minsup$, alors α est un *motif séquentiel* fréquent. L'objectif de la recherche de motifs séquentiels est donc d'extraire l'ensemble complet des motifs séquentiels fréquents par rapport à un seuil de support minimum $minsup$.

L'algorithme GSP (Generalized Sequential Patterns) [SA96a] s'appuie sur l'antimonotonie du support (le support d'une super-séquence est inférieur ou égal au support de toutes ses sous-séquences). Cet algorithme est basé sur le paradigme *Apriori*. Cet algorithme permet l'extraction de motifs séquentiels en effectuant plusieurs passes sur les données avec une approche de génération et validation de candidats. GSP généralise également la définition [AS95] en incluant des contraintes de temps, des fenêtres glissantes, et des hiérarchies définies par l'utilisateur.

L'algorithme PSP (Prefix-tree for Sequential Patterns) [MCP98] s'appuie sur un arbre des préfixes afin d'améliorer la génération de candidats. L'algorithme SPAM [AFGY02] parcourt en profondeur l'espace de recherche et représente la base de données sous formes de vecteurs de bits, ce qui permet un calcul du support efficace.

L'algorithme SPADE [Zak01] utilise une représentation verticale de la base de données pour extraire des motifs séquentiels. Cette approche est une extension des travaux d'extraction d'itemsets fréquents CHARM [ZH02]. Dans les bases de données verticales, la base de données devient un ensemble de n-uplets de la forme $\langle itemset : (sequence\_ID, event\_ID)\rangle$. L'ensemble des paires ID d'un itemset donné forme l'identifiant de la liste (ID_list) de l'itemset. Pour découvrir les k-séquences (séquences contenant k items), l'algorithme SPADE joint les ID_lists de deux éléments de l'ensemble des $k-1$-séquences fréquentes. La longueur de la liste résultante est égale au support de la k-séquence générée. La procédure s'arrête quand aucune séquence fréquente ne peut être générée ou qu'aucune séquence ne peut être jointe. L'utilisation de base de données verticales permet d'améliorer l'étape de vérification des séquences candidates.

L'algorithme PrefixSpan [PHMA+01, PHMA+04] permet l'extraction de motifs séquentiels à l'aide du paradigme *pattern growth*. Cet algorithme utilise une méthode *diviser pour régner*. Le premier passage sur la base de données permet d'extraire l'ensemble des 1-séquences fréquentes. Chaque motif séquentiel est considéré comme un *préfixe*. L'ensemble complet des motifs séquentiels est ainsi partitionné en différents sous-ensembles par rapport à différents préfixes. Pour extraire les sous-ensembles de motifs séquentiels, des *bases de données projetées* sont construites et fouillées récursivement.

Deux approches d'extraction de motifs séquentiels *clos* ont été définies à partir du paradigme *pattern growth* [YHA03, WHL07].

L'algorithme DISC [CWC04] propose d'extraire les séquences fréquentes sans calculer le support des séquences non fréquentes. Pour cela, l'algorithme DISC supprime les séquences non fréquentes en les comparant aux séquences de même longueur.

Dans [YWK07], les auteurs proposent l'algorithme LAPIN afin d'extraire de longues séquences. Cet algorithme étudie la dernière position d'un item s afin de juger si une k-séquence fréquente peut être étendue avec l'item s pour former une $k+1$-séquence. Une amélioration de LAPIN est proposée avec l'algorithme PAID [YKW06] qui s'appuie sur une représentation verticale de la base de données.

L'algorithme MSPX, défini par [LC05], propose d'extraire les motifs séquentiels maximaux en utilisant plusieurs échantillons de la base, ce qui permet d'exclure plus rapidement les séquences candidates non fréquentes.

MEMISP (MEMory Indexing for Sequential Patterns) [LL02] ne nécessite pas de multiples passes sur la bases de données ou la génération de bases de données intermédiaires. Cet algorithme ne requiert qu'une seule passe sur la base de données, au plus deux pour les bases de données très importantes. De plus, il évite de générer des candidats et de construire des bases de données projetées. Il s'appuie sur une

recherche récursive et une stratégie d'indexation afin de générer tous les motifs séquentiels dans la base de données stockée en mémoire.

Parallèlement au motifs séquentiels des travaux similaires se sont développés. Dans [MTV97], les auteurs s'intéressent aux *épisodes* fréquents dans les séquences de données. Les épisodes sont des graphes orientés sans cycle où les nœuds sont des événements et les arêtes décrivent la relation de précedence entre les événements.

L'approche SPIRIT [GRS99] permet de cibler la recherche de motifs séquentiels sur ceux qui sont jugés intéressants par l'utilisateur. Les choix de l'utilisateur sont indiqués à l'aide d'expressions régulières. Cette méthode présente l'avantage de se ne pas perdre de temps à l'extraction de motifs jugés inintéressants par l'utilisateur.

Les motifs séquentiels permettent d'extraire des motifs qui prennent en compte le temps. Seul l'approche de [SA96a] permet de prendre en compte les hiérarchies. Très peu de travaux proposent de prendre en compte des valeurs numériques.

L'approche définie dans [KLNS04] utilise des intervalles afin d'extraire des séquences fréquentes dans lesquelles les items sont associés à des quantités. Les motifs séquentiels extraits sont alors des séquences dont les items sont des couples (item, quantité maximale). L'algorithme se divise en deux étapes. Tout d'abord, les items sont extraits sans tenir compte des quantités. Ensuite, pour chaque item, on calcule la quantité maximale associée à l'item.

Des approches issues de la logique floue ont été proposées pour définir les *motifs séquentiels flous*. Ces propositions permettent d'extraire des motifs séquentiels à partir de données quantitatives [CH03, CH06, CTCH01, HLW01, HTC04, SG05, FLT07]. Toutes ces approches sont fondées sur le même principe. Afin d'extraire les motifs, les univers des quantités de chaque attribut sont partitionnés en plusieurs sous-ensembles flous. Le jeu de données est ensuite converti en une base de degrés d'appartenance. Ces approches diffèrent ensuite dans le calcul du support d'une séquence.

5 Motifs séquentiels multidimensionnels

L'extraction de motifs séquentiels revient à extraire des corrélations au sein d'un seul attribut au cours d'une relation d'ordre (i.e. temps). Considérer plusieurs attributs ou dimensions peut permettre la découverte de motifs qui décrivent mieux les données et qui peuvent être plus utiles pour les utilisateurs. Par exemple, en considérant plusieurs dimensions, il est possible de découvrir des comportements différents en fonction des groupes de consommateurs. Par exemple, les *étudiants* achètent fréquemment le produit A puis le produit B une semaine après. La séquence $\langle A, B\rangle$ n'est supportée fréquemment que par les étudiants et par aucun autre groupe.

Les motifs séquentiels multidimensionnels ont été définis pour la première fois par [PHP+01]. Dans l'extraction de motifs séquentiels multidimensionnels, différents attributs de l'identifiant de la transaction sont introduits et forment un ensemble de séquences multidimensionnelles comme illustré dans le tableau

Tab. 1.1. Le but de cette extraction est de découvrir des motifs séquentiels plus intéressants en prenant en compte plusieurs attributs ou dimensions. Dans [PHP+01], une extension directe de PrefixSpan [PHMA+04] est introduite ainsi qu'une combinaison des algorithmes PrefixSpan et BUC (Bottom Up Computation of iceberg cube) [BR99].

cid	Cust-Grp	City	Age-grp	product-sequence
10	business	Boston	middle	$\langle(bd)cba\rangle$
20	professional	Chicago	young	$\langle(bf)(ce)(fg)\rangle$
30	business	Chicago	middle	$\langle(ah)abf\rangle$
40	education	New York	retired	$\langle(be)(ce)\rangle$

TAB. 1.1 – Base de séquences multidimensionnelles

Dans la première approche, appelée UniSeq (Unique Sequence), les attributs multidimensionnels sont associés aux séquences de la base de séquences comme illustré dans le tableau Tab. 1.2. PrefixSpan est ensuite utilisé. Il extrait toutes les 1-séquences en un passage sur les données en comptant le nombre d'occurrences de chaque item. Pour toutes les 1-séquences fréquentes, la base de données projetée est construite et fouillée récursivement.

cid	Extension multidimensionnelle des séquences
10	$\langle(business, Boston, middle)(bd)cba\rangle$
20	$\langle(professional, Chicago, young)(bf)(ce)(fg)\rangle$
30	$\langle(business, Chicago, middle)(ah)abf\rangle$
40	$\langle(education, New\ York, retired)(be)(ce)\rangle$

TAB. 1.2 – Extension multidimensionnelle de la base de données séquentielles

Une autre approche, Seq-Dim, partitionne la base de données en deux. Une partie identifie les attributs multidimensionnels alors que la seconde partie contient les séquences définies sur une seule dimension. Ainsi, le problème de l'extraction de motifs séquentiels multidimensionnels peut se décomposé en deux sous-problèmes : extraire des motifs multidimensionnels et extraire des motifs séquentiels.

Il y a deux façons de résoudre ce problème. Dans Dim-Seq, les motifs multidimensionnels sont extraits en premier, et pour chaque motif multidimensionnel, la base de données projetée est construite et les motifs séquentiels y sont extraits. Dans Seq-Dim, les motifs séquentiels sont d'abord extraits, puis pour chaque motif séquentiel, l'extraction de motifs multidimensionnels est appliquée sur les bases de données projetées.

Des expérimentations comparent les trois algorithmes UniSeq, Dim-Seq, et Seq-Dim. Ces résultats montrent que Seq-Dim est le plus performant dans la plupart des cas. Quand le nombre de dimensions est faible, l'algorithme UniSeq est le plus rapide.

Plusieurs propositions découlent directement de la définition des motifs séquentiels multidimensionnels de [PHP+01].

Dans [RKK07], les auteurs proprosent une application de [PHP+01] dans un contexte de télécommunications. Ils proposent un algorithme appelé MobilePrefixSpan afin d'extraire des motifs décrivant les mouvements des utilisateurs de mobiles. Toutefois, les auteurs ne considèrent que les ordres consécutifs dans leur problématique, c'est-à-dire des événements qui se produisent consécutivement (sans d'autres événements entre eux).

Dans [SZ05a], les auteurs montrent l'utilité des motifs séquentiels multidimensionnels pour étudier les comportements des internautes (web usage mining). De plus, les auteurs proposent que les attributs multidimensionnels puissent être numériques ou symboliques. La présence de valeurs numériques nécessite une gestion particulière. Ainsi une séquence multidimensionnelle est supportée par une séquence de données si elle est suffisamment similaire à la séquence de données. Néanmoins, aucun algorithme n'est défini dans cette proposition.

Dans [JHC+07], les auteurs proposent un algorithme nommé ApproxMGMSP qui permet d'extraire des motifs séquentiels multidimensionnels, comme défini dans [PHP+01], par approximation des systèmes distribués. L'approximation de l'extraction des séquences multidimensionnelles s'appuie sur l'utilisation de la distance de Levenshtein.

La proposition de [PHP+01] permet ainsi d'obtenir des motifs séquentiels multidimensionnels fréquents *intra motif multidimensionnels* puisque chaque séquence fréquente est *contextualisé* par un motif multidimensionnel fréquent associé. Remarquons que la proposition de [PHP+01] ne permet pas d'extraire des motifs multidimensionnels *inter motifs multidimensionnels*. Le motif multidimensionnel ne peut pas varier au cours de la séquence. Nous souhaitons obtenir des corrélations entre toutes les dimensions, c'est-à-dire extraire des séquences multidimensionnelles inter motifs multidimensionnels.

Dans [YC05], les auteurs tentent d'étendre la recherche de motifs séquentiels au contexte des bases de données décrivant les informations au moyen de plusieurs attributs. Cependant, cette approche est restreinte au cas particulier où les dimensions étudiées entretiennent entre elles un lien hiérarchique. Ainsi, dans l'exemple pris par les auteurs, les différentes dimensions sont liées au comportement d'internautes dont les visites de pages sont organisées en transactions (dimension 1) elles-mêmes organisées en sessions (dimension 2) elles-mêmes organisées en jours (dimension 3). Ces différentes dimensions sont imbriquées au sein des motifs trouvés et il est impossible de retrouver les valeurs fréquentes le long de ces dimensions, celles-ci n'intervenant que pour organiser le temps de manière hiérarchique. De même que dans l'approche proposée par [PHP+01], les séquences ne concernent qu'une seule dimension (les pages internet dans ce cas).

L'approche de [dAFGL04] est basée sur la logique temporelle du premier ordre. Cette proposition est présente des limites non négligeables. La notion de groupe n'est pas codée dans la base mais déterminée par l'utilisateur et plusieurs attributs peuvent apparaître dans les séquences. Concernant ce dernier point,

	Relation d'ordre	Multidimensionnalité	Hiérarchies	Valeurs numériques
Règles d'association multidimensionnelles		•	[IKA02, DHL+01] [NJ03, TT04]	•
Motifs colossaux		•		
Motifs séquentiels	•		[SA96a]	[KLNS04, CH03] [CTCH01, HLW01] [HTC04, SG05] [FLT07, CH06]
Motifs séquentiels multidimensionnels	•	•	[YC05]	[SZ05a]

TAB. 1.3 – Synthèse des approches

les auteurs mentionnent la possibilité de traiter plusieurs attributs, cependant le formalisme mis en place n'est pas étendu de manière complète au cas multi-attributs.

6 Discussion

Le tableau 1.3 décrit les différentes approches d'extraction de motifs présentés précédemment en fonction des critères d'analyse définis (prise en compte du temps, des hiérarchies, de la multidimensionnalité et des valeurs numériques).

Les techniques d'extraction de règles d'association multidimensionnelles proposent de prendre en compte la multidimensionnalité inhérente aux bases de données qu'elles analysent. Deux types de corrélations peuvent être extraits. Des approches proposent d'établir des corrélations entre des valeurs différentes d'une même dimension (intra dimension) comme dans le contexte classique d'extraction de règles d'association [AIS93] où les corrélations sont extraites entre les valeurs de la dimension *produits*. Des corrélations entre les valeurs de différentes dimensions sont également recherchées (i.e les *étudiants* achète fréquemment le produit A). Des travaux proposent de regrouper ces deux types de règles en une seule : les règles hybrides. Certaines approches s'attaquent à la prise en compte des hiérarchies. Puisque l'ambition de ces travaux est d'extraire des règles d'association dans des bases de données multidimensionnelles, nous pouvons supposer qu'ils prennent tous en compte la ou les dimensions numériques présentes servant à représenter *la mesure*.

Les motifs colossaux s'attaquent à la découverte de motifs dans des données très spécifiques (microarrays) qui sont définies sur un nombre extrêmement important de dimensions et dont les cardinalités sont relativement faibles. Du fait de leur problématique, ces travaux ne s'intéressent pas à la prise en

compte du temps ou des hiérarchies. Pour les valeurs numériques, les algorithmes sont exécutés sur des fichiers prétraités. Par exemple, l'expression des gènes est discrétrisée (sur-exprimé, sous-exprimé, etc.).

Les motifs séquentiels ont été proposés pour extraire des corrélations entre événements suivant une relation d'ordre (i.e. temps) [AS95]. Ils tentent de mettre en exergue des corrélations entre des valeurs d'une même dimension au cours du temps. A notre connaissance, seule l'approche de [SA96a] permet de prendre en compte des séquences définies sur plusieurs niveaux de hiérarchies. Pour la prise en compte de valeurs numériques, plusieurs travaux, essentiellement basés sur la logique floue, proposent d'associer une quantité à chaque item d'une séquence.

Les motifs séquentiels ne permettent pas la prise en compte de la multidimensionnalité. C'est pourquoi les motifs séquentiels multidimensionnels ont été définis dans [PHP+01]. Cette approche propose d'extraire des séquences fréquentes définies sur une seule dimension et de les caractériser par un motif multidimensionnel défini sur les autres dimensions. Le motif multidimensionnel caractérisant la séquence fréquente ne varie pas au cours du temps. La prise en compte du temps n'est donc relative qu'à une seule dimension. De plus, cette proposition et celles qui en découlent ne permettent pas de prendre en compte les hiérarchies ou les valeurs numériques.

L'approche de [SZ05a] propose de prendre en compte des attributs numériques à partir de la définition de [PHP+01] en introduisant des mesures de similarité pour calculer le support d'une séquence. Toutefois, cette proposition n'est étayée par aucune définition ou algorithme permettant de répondre aux motivations des auteurs.

L'approche de [YC05] propose d'extraire des séquences dans des données multidimensionnelles. Dans cette proposition, les hiérarchies sont utilisées pour représenter le temps de manière plus fine. En effet, un internaute visite une *page web* durant *session* au cours d'une journée. Ces trois dimensions entretiennent un très fort lien hiérarchique entre elles ($Jour > Session > Page$). Nous ne pouvons pas dire que cette approche traite réellement la multidimensionnalité et les hiérarchies. Des plus, les auteurs ne s'attaquent pas à la prise en compte de valeurs numériques.

L'approche de [dAFGL04] propose une approche basée sur la logique temporelle du premier ordre pour extraire des motifs séquentiels définis sur plusieurs dimensions. Les auteurs ne proposent ni la prise en compte des hiérarchies ni la prise en compte des valeurs numériques.

A notre connaissance, il n'existe donc aucune proposition tentant d'extraire des corrélations entre plusieurs dimensions au cours d'une relation d'ordre (i.e. temps) ni de prendre en compte les hiérarchies et les valeurs numériques.

Première partie

Motifs Séquentiels : d'Une à Plusieurs Dimensions

La science est le capitaine, et la pratique, ce sont les soldats.

Léonard de Vinci —

De nombreux jeux de données de la vie réelle sont définis suivant plusieurs dimensions ou attributs. Dans le chapitre précédent, nous avons vu qu'il n'existe pas d'approche proposant d'extraire des motifs où la temporalité se manifeste sur toutes les dimensions considérées. Par exemple, la définition introduite par [PHP+01] ne considère le temps que sur une seule dimension. Les séquences fréquentes ainsi extraites sont alors associées à des motifs multidimensionnels qui sont « invariables » en fonction du temps.

Cette observation souligne la nécessite de proposer une nouvelle définition des motifs séquentiels multidimensionnels que nous proposons dans le chapitre 1. Nous montrons également que le problème d'extraction associé est une généralisation des problèmes d'extraction de motifs séquentiels classiques [AS95] et des motifs de [PHP+01]. L'espace de recherche associé à ce problème est très important puisqu'il combine à la fois multidimensionnalité et temporalité. Nous proposons deux manières différentes d'extraire des motifs séquentiels multidimensionnels.

Dans le chapitre 2, nous ciblons l'espace de recherche en extrayant des motifs séquentiels multidimensionnels composés des items les plus spécifiques avec l'algorithme M^2SP.

Dans le chapitre 3, nous proposons d'élaguer efficacement l'espace de recherche à l'aide de représentation condensée. Nous définissons les motifs séquentiels multidimensionnels clos qui permettent à la fois d'introduire des propriétés supplémentaire d'élagage de l'espace de recherche et de représenter l'ensemble des motifs extraits de manière non redondante. Deux algorithmes, $CMSP\_Cand$ et $CMSP\_Free$ sont définis.

Enfin, nous terminons cette partie par une discussion des perspectives de recherche associées à ces propositions.

Chapitre 1

Motifs Séquentiels Multidimensionnels

1.1 Introduction

Les motifs séquentiels sont apparus afin de permettre la découverte de règles intégrant la notion de temporalité et d'enchaînement d'événements [AS95]. De telles règles seront par exemple de la forme : *les clients qui ont acheté un téléviseur et un lecteur DVD achètent plus tard un magnétoscope numérique*.

De nombreux travaux ont traité cette problématique. Les recherches se sont notamment intéressées à l'extraction efficace des motifs face à de gros volumes de données [AS95, AFGY02, PHMA$^+$04, CWC04, MCP98, Zak01] avec de nombreuses applications à la clé [HLC01, YWYH02, TC90, TTM04].

Toutefois, les propositions existantes ne travaillent que sur une seule dimension d'analyse, nommée $produit$ dans les approches de type *étude du panier de la ménagère*. Ainsi, même si cette dimension peut être modifiée dans des applications de recherche de motifs séquentiels à d'autres domaines que le panier de la ménagère (par exemple dans le cadre de l'étude des comportements d'internautes [TTM04]), il n'en reste pas moins qu'il n'est possible d'analyser qu'une seule dimension à la fois.

Ainsi, il n'existe pas à l'heure actuelle de méthode permettant de mettre en exergue des corrélations entre valeurs de différents attributs, par exemple pour découvrir des règles de la forme $\langle\{$(surf,NY),(housse,NY)$\},\{$(combi,SF)$\}\rangle$ indiquant qu'un nombre suffisant (au sens du support) de personnes ont acheté leur planche de surf et la housse à New York avant de se rendre à San Francisco où ils ont acheté une combinaison.

Si la littérature recense des contributions liées aux motifs séquentiels multidimensionnels proposées par [PHP$^+$01], celles-ci ne permettent pas de combiner plusieurs attributs au sein des motifs extraits pour ce qui est de la partie séquentielle, les multiples attributs n'apparaissant que pour qualifier ou restreindre le cadre dans lequel on trouve la séquence fréquente.

Il est donc primordial de proposer une nouvelle définition des motifs séquentiels multidimensionnels ou multi-attributs afin de permettre à toutes les dimensions de « varier » au cours du temps.

Dans ce chapitre, nous proposons une nouvelle définition des motifs séquentiels multidimensionnels ou multi-attributs.

Ce chapitre s'organise de la manière suivante. La section 1.2 introduit la définition des motifs séquentiels multidimensionnels. Dans la section 1.3, nous montrons que le problème d'extraction de motifs séquentiels multidimensionnels que nous avons défini est une généralisation de problèmes existants. Enfin, la section 1.4 conclut et présente les principales perspectives associées à ce chapitre.

1.2 Motifs séquentiels multidimensionnels

Dans cette section, nous présentons une nouvelle définition des motifs séquentiels multidimensionnels.

1.2.1 Données manipulées

Nous étendons les concepts présentés précédemment (client - date - items) en considérant non plus des attributs simples pour décrire les données, mais des ensembles d'attributs.

Nous supposons qu'il existe au moins une dimension (*e.g.* temporelle) dont le domaine est totalement ordonné.

Définition 1.1 (Partition des dimensions)**.** Pour chaque table relationnelle définie sur un ensemble de dimensions $\mathcal{D}$, nous considérons une partition de $\mathcal{D}$ en quatre sous-ensembles notés respectivement :

- D_R pour l'ensemble des dimensions de référence (client dans le contexte classique) qui permettent de déterminer si une séquence est fréquente.
- D_T pour l'ensemble des dimensions permettant d'introduire une relation d'ordre.
- D_A pour l'ensemble des dimensions d'analyse (produit dans contexte classique) sur lesquelles les corrélations seront extraites.
- D_I pour l'ensemble des dimensions ignorées.

Il en découle que chaque n-uplet $c = (d_1, \ldots, d_n)$ peut s'écrire sous la forme d'un quadruplet $c = (i, r, a, t)$ où i, r, a et t sont respectivement les projections de c sur D_I, D_R, D_A et D_T.

Définition 1.2 (Bloc)**.** Etant donnée une table relationnelle DB, on appelle bloc l'ensemble des n-uplets de DB qui ont la même projection r sur D_R.

Chaque bloc B_r est identifié par le n-uplet r qui le définit. On note B_{DB,D_R}, l'ensemble des blocs de DB définis à partir des dimensions de référence D_R.

Exemple 1.1. Nous considérons la partition suivante des dimensions de la base de données DB représentée par le tableau Tab. 1.1 :

- $D_I = \emptyset$
- $D_R = \{Cust\text{-}Grp,\ City\ \}$

- $D_A = \{Age,\ Product\}$
- $D_T = \{Date\}$

Il est possible de diviser DB en trois blocs définis par les n-uplets $(Educ, Chicago)$ (cf tableau 1.2), $(Educ, Los\ Angeles)$ (cf tableau 1.3) et $(Reti., Miami)$ (cf tableau 1.4). Ces trois blocs définissent une partition en blocs de DB.

Date	Cust-Grp	City	Age	Product
1	Educ	Chicago	A	clou
1	Educ	Chicago	B	pneu
1	Educ	Los Angeles	A	clou
1	Reti.	Miami	C	clou
1	Reti.	Miami	C	marteau
2	Educ	Chicago	B	rustine
2	Educ	Chicago	B	pneu
2	Educ	Los Angeles	A	clou
3	Educ	Los Angeles	B	rustine

TAB. 1.1 – Base de données DB

Date	Cust-Grp	City	Age	Product
1	Educ	Chicago	A	clou
1	Educ	Chicago	B	pneu
2	Educ	Chicago	B	pneu
2	Educ	Chicago	B	rustine

TAB. 1.2 – $B_{(Educ,Chicago)}$

Date	Cust-Grp	City	Age	Product
1	Educ	Los Angeles	A	clou
2	Educ	Los Angeles	A	clou
3	Educ	Los Angeles	B	rustine

TAB. 1.3 – $B_{(Educ,LosAngeles)}$

Dans le cadre de l'extraction des motifs séquentiels multidimensionnels, l'ensemble D_R permet d'identifier les blocs par rapport auxquels le support sera calculé. Pour cette raison, cet ensemble est nommé *référence*. On note que dans le cadre des motifs séquentiels classiques et des extensions [PHP+01] et [dAFGL04], cet ensemble est réduit à un seul attribut (identifiant du client cid du tableau Tab. **??** ou identifiant IdG dans [dAFGL04]). Dans nos calculs, le support d'une séquence sera ainsi calculé comme étant la proportion de blocs où cette séquence peut être retrouvée.

Date	Cust-Grp	City	Age	Product
1	Reti.	Miami	C	clou
1	Reti.	Miami	C	marteau

TAB. 1.4 – $B_{(Reti.,Miami)}$

L'ensemble D_A décrit les axes d'*analyse*, c'est-à-dire l'ensemble des dimensions apparaissant explicitement dans les motifs séquentiels multidimensionnels extraits. Dans le cadre des motifs séquentiels classiques, seule une dimension apparaît, correspondant aux produits achetés (ou encore aux pages internet visitées). Dans notre approche, cette dimension est étendue à la prise en compte d'un ensemble de dimensions.

L'ensemble D_I décrit les axes *ignorés*, c'est-à-dire ceux qui ne servent ni à définir la date, ni à identifier un sous-cube, ni à définir le motif lui-même.

1.2.2 Item, itemset et séquence multidimensionnels

Etant donnée une base de données DB, nous considérons une partition $\{D_R, D_A, D_T, D_I\}$ de l'ensemble des dimensions $\mathcal{D}$ ($|D_A|$=m). Nous pouvons maintenant définir les concepts fondamentaux d'item, itemset et séquence dans un contexte multidimensionnel.

Définition 1.3 (Item multidimensionnel)**.** Un item multidimensionnel $e = (d_1, \ldots, d_m)$ est un m-uplet défini sur les dimensions d'analyse D_A tel que :

- $\forall i \in [1, \ldots\ m],\ d_i \in Dom(D_i) \cup \{*\}$ et $D_i \in D_A$
- $\exists d_i \in [1, |dots, m]$ t.q. $d_i \neq *$

$(A, clou)$, $(B, pneu)$ et $(*, rustine)$ sont des items multidimensionnels.

Il peut être très difficile de trouver une instantiation sur l'intégralité des dimensions d'analyse fréquente. Pour cette raison, nous introduisons une valeur *joker* symbolisée par $*$. Cette valeur signifie que l'on ne tient pas compte de la valeur sur la dimension d'analyse ciblée et permet ainsi de réduire ponctuellement le nombre de dimensions d'analyse.

Un item multidimensionnel doit contenir au moins une dimension d'analyse spécifiée (non étoilée). En effet, découvrir l'item $(*, *, \ldots, *)$ n'a aucun intérêt. La fréquence de cet item est $|B_{DB,D_R}|$.

Deux items multidimensionnels peuvent être comparables. Ainsi un item e peut être inclus dans un item e' d'après la définition suivante :

Définition 1.4. Un item multidimensionnel $e = (d_1, \ldots, d_m)$ est inclus dans un item multidimensionnel $e' = (d'_1, \ldots, d'_m)$ (noté $e \subseteq e'$) si $\forall i \in [1, \ldots, m]\ d_i = d'_i$ ou $d'i = *$.

Définition 1.5 (Itemset multidimensionnel)**.** Un itemset multidimensionnel $i = \{e_1, e_2, \ldots, e_p\}$ est un ensemble non vide d'items multidimensionnels.

Il est important de remarquer que d'une part tous les items d'un itemset sont définis sur les mêmes dimensions (D_A), et que d'autre part les items multidimensionnels d'un même itemset sont deux à deux distincts. Ainsi, $\{(A, clou), (B, pneu)\}$ et $\{(B, rustine)\}$ sont des itemsets multidimensionnels alors que $\{(A, clou), (A, *)\}$ n'en est pas un puisque l'item $(A, clou)$ est inclus dans l'item $(A, *)$ (noté $(A, clou) \subseteq (A, *)$). On dit aussi que $(A, clou)$ est plus spécifique que $(A, *)$.

Un itemset i peut également être inclus dans un itemset i' si pour tous les items e de i, il existe des items distincts e' de i' tels que $e \subseteq e'$. Par exemple, l'itemset $i = \{(A, clou)\}$ est inclus dans l'itemset $i' = \{(A, *), (B, pneu)\}$ ($i \subseteq i'$).

Définition 1.6 (Séquence multidimensionnelle). Une séquence multidimensionnelle $\varsigma = \langle i_1, \ldots, i_l \rangle$ est une liste ordonnée non vide d'itemsets multidimensionnels.

$\langle \{(A, clou), (B, pneu)\}\ \{(B, rustine)\} \rangle$ est une séquence multidimensionnelle.

Une séquence peut être incluse dans une autre d'après la définition suivante.

Définition 1.7 (Inclusion de séquence). Une séquence multidimensionnelle $\varsigma = \langle a_1, \ldots, a_l \rangle$ est appelée sous-séquence d'une séquence $\varsigma' = \langle b_1, \ldots, b_{l'} \rangle$ ($\varsigma \prec \varsigma'$) s'il existe des entiers $1 \leq j_1 \leq j_2 \leq \ldots \leq j_l \leq l'$ tels que $a_1 \subseteq b_{j_1}, a_2 \subseteq b_{j_2}, \ldots, a_l \subseteq b_{j_l}$.
On dit aussi que la séquence ς' est une super-séquence de ς.

Soient les séquences multidimensionnelles $\varsigma = \langle \{(A, clou)\}, \{(B, rustine)\} \rangle$ et $\varsigma' = \langle \{(A, *), (B, pneu)\}$,-$\{(B, rustine)\} \rangle$, ς est une sous-séquence de ς'.

1.2.3 Support

Calculer le support d'une séquence revient donc à compter le nombre de blocs de la base qui la supportent, de même qu'il revient à compter le nombre de clients ayant suivi la séquence d'achat dans le contexte du panier de la ménagère.

Définition 1.8 (Séquence et Bloc). Un bloc de données B_r supporte une séquence $\varsigma = \langle i_1, \ldots, i_l \rangle$ si :
- $\forall j = 1 \ldots l$, $\exists d_j \in Dom(D_t)$, $\forall e = (a_{i_1}, \ldots, a_{i_m}) \in i_j$, $\exists c = (i, r, (x_{i_1}, \ldots, x_{i_m}), d_j) \in B_r$ avec $(x_{i_1}, \ldots, x_{i_m}) \subset e$
- $d_1 < d_2 < \cdots < d_l$.

Un bloc supporte une séquence si on peut trouver un ensemble de cellules vérifiant chacun des itemsets de la séquence en respectant les contraintes de temps imposées par l'ordre des itemsets. La présence d'une valeur joker $*$ sur une dimension d'un item multidimensionnel équivaut à relâcher une contrainte lorsqu'on souhaite appairer l'item avec un n-uplet d'un bloc.

Il est important de remarquer que même si un bloc supporte une séquence, celle-ci n'est pas nécessairement incluse dans le bloc. En effet, dès qu'il y a une * dans un item de la séquence, l'inclusion

(Définition 1.7 n'est plus vérifiée. Ceci est dû au fait que "tout est inclus dans *", et cela se répercute au niveau des items, itemsets et des séquences.

Par exemple, le bloc $B_{(Educ,Chicago)}$ du tableau Tab. 1.2 supporte la séquence $\langle\{(A, clou), (*, pneu)\}$,-$\{(*, pneu)\}\rangle$ puisque l'on retrouve bien la date 1 à laquelle l'itemset $\{(A, clou), (*, pneu)\}$ est vérifié et la date 2 à laquelle l'itemset $\{(*, pneu)\}$ est vérifié. Or cette séquence n'est pas incluse dans la séquence de données identifiée par $B_{(Educ,Chicago)}$.

On peut exprimer le support d'une séquence de deux façons différentes. Ainsi le *support absolu* d'une séquence multidimensionnelle correspond donc au nombre de blocs définis sur D_R qui supportent cette séquence.

$$\text{support\_absolu}(\varsigma) = |B_r \in B_{DB,D_R}\ t.q.\ B_r \text{ supporte } \varsigma|$$

Le *support relatif*, quant à lui, correspond à la proportion de blocs de B_{DB,D_R} qui supportent la séquence.

$$\text{support\_relatif}(\varsigma) = \frac{\text{support\_absolu}(\varsigma)}{|B_{DB,D_R}|}$$

Définition 1.9 (Séquence fréquente)**.** Soit σ, le seuil de support minimum fixé par l'utilisateur, une séquence ς est dite fréquente si $support(\varsigma) \geq \sigma$.

Exemple 1.2. On considère la base de données du tableau Tab. 1.1 avec $\mathscr{D}_{\mathscr{R}} = \{$Cust-Grp, City$\}$, $\mathscr{D}_{\mathscr{A}} = \{Age, Product\}$, $\sigma = 1$. On cherche le support de la séquence $\varsigma = \langle\{(A, clou), (B, pneu)\}, \{(B, rustine)\}\rangle$. Ce calcul nécessite le parcours de tous les blocs de B_{DB,D_R}, c'est-à-dire les trois blocs illustrés par les tableaux 1.2, 1.3 et 1.4.

1. $B_{(Educ,Chicago)}$ (cf tableau Tab. 1.2). Dans ce bloc, deux dates sont présentes. A la date 1, on a bien $(A, clou)$ et $(B, pneu)$. Puis à la date 2 on retrouve bien $(B, rustine)$. Donc ce bloc supporte la séquence ς.
2. $B_{(Educ,LosAngeles)}$ (cf tableau Tab. 1.3). Ce bloc ne supporte pas l'item $(B, pneu)$, et donc pas la séquence ς.
3. $B_{(Reti.,Miami)}$ (cf tableau Tab. 1.4). Ce bloc ne contient qu'une seule date, il ne peut donc pas supporter la séquence ς qui nécessite des blocs contenant au moins deux dates différentes.

On a donc : $support(\varsigma) = 1 \geq \sigma$, et par conséquent, ς est une séquence fréquente.

Le problème de l'extraction des motifs séquentiels multidimensionnels est exposé dans la définition suivante.

Définition 1.10 (Extraction des motifs séquentiels multidimensionnels)**.** Etant donnés un ensemble de n-uplets définis sur un ensemble de dimensions $\mathcal{D}$, la partition de $\mathcal{D}$ en quatre sous-ensembles D_A, D_R, D_T et D_I, un seuil de support minimum σ, le but de l'extraction de motifs séquentiels multidimensionnels est de découvrir l'ensemble complet des séquences fréquentes.

1.3 Propriétés des Motifs Séquentiels Multidimensionnels

Dans cette section, nous montrons que le problème de l'extraction des motifs séquentiels multidimensionnels (définition 1.10) est une généralisation des problèmes d'extraction de motifs séquentiels classiques [AS95] et de la proposition de [PHP+01].

Proposition 1 (Généralisation des Motifs Séquentiels)**.** Le problème d'extraction de motifs séquentiels multidimensionnels est une généralisation du problème d'extraction de motifs séquentiels [AS95].

Démonstration. La recherche de motifs séquentiels "monodimensionnels" vise à extraire des séquences au sein de données qui sont décrites par un identifiant de client cid, un identifiant de date $date\_id$ et des items. Pour se ramener au même problème, nous devons imposer plusieurs contraintes dans le choix des différents axes :

- L'ensemble des dimensions d'analyse D_A doit être réduit à une seule dimension (*e.g.* la dimension produit).
- L'ensemble des dimensions de référence est réduit à une seule dimension (*e.g.* cid).
- L'ensemble des dimensions doit être réduit à une seule dimension (*e.g.* $date\_id$).

Ainsi, D_A, D_R et D_T sont des singletons, des séquences définies sur D_A seront recherchées. Nous pouvons remarquer qu'il ne peut y avoir de valeur joker * puisque $|D_A| = 1$ et qu'un item doit avoir, par définition, une valeur non-étoilée. Etant donné que $|D_R| = 1$, le support d'une séquence correspond au nombre de clients (blocs) qui supportent la séquence. □

Proposition 2 (Généralisation de [PHP+01])**.** Le problème d'extraction de motifs séquentiels multidimensionnels est une généralisation du problème introduit dans [PHP+01].

En d'autres termes, notre définition permet d'extraire tous les motifs définis dans [PHP+01] et d'extraire également des motifs qui ne peuvent pas être extraits en se référant à la définition de [PHP+01].

Démonstration. Soit $\varsigma = (p, s)$ un motif séquentiel multidimensionnel relatif à [PHP+01] où s est une séquence d'items. Par définition, ς est fréquent si $p = (d_1, \ldots, d_{m-1})$ est un motif fréquent et s une séquence fréquente.

Posons $s = \langle i_1, \ldots, i_l \rangle$ où $i_i = \{e_{i_1}, \ldots, e_{i_j}\}$. Comme ς est fréquent, pour chaque itemset i_k on a :

- $i'_1 = \{(p, e_{1_1}), \ldots, (p, e_{1_j})\}$ fréquent,
- ...
- $i'_k = \{(p, e_{k_1}), \ldots, (p, e_{k_j})\}$ fréquent,
- ...
- $i'_l = \{(p, e_{l_1}), \ldots, (p, e_{l_j})\}$ fréquent.

(p, i_k) et i'_k décrivent la même information de manière différente. Donc, tous les itemsets de [PHP+01], sont retrouvés avec notre définition. On suppose que les deux problèmes d'extractions de motifs séquentiels multidimensionnels se basent sur la même relation d'ordre. Ainsi, par rapport à notre définition, la

séquence $\varsigma' = \langle i'_1, \ldots, i'_l \rangle$ est un motif séquentiel multidimensionnel. ς et ς' décrivent la même information. Par conséquent, tous les motifs séquentiels multidimensionnels selon la définition de [PHP+01] sont également des motifs séquentiels multidimensionnels par rapport à notre définition. Notre proposition permet également de définir des motifs séquentiels multidimensionnels qui ne peuvent pas être extraits avec [PHP+01] : les motifs *inter motifs multidimensionnels* (plusieurs dimensions d'un motifs sont susceptibles de varier au cours la relation d'ordre, ex : $\langle \{(A, clou), (B, pneu)\}\{(B, rustine)\}\rangle$). □

1.4 Discussion

Dans ce chapitre, nous proposons une définition des motifs séquentiels multidimensionnels afin d'extraire des corrélations entre des événements définis sur plusieurs dimensions selon leur chronologie d'apparition. Cette définition permet ainsi une meilleure appréhension des données sources puisque les corrélations sont désormais extraites sur plusieurs dimensions. De plus, la partition de l'ensemble des dimensions permet à l'utilisateur une plus grande interaction avec les données en lui permettant de choisir les différents sous-ensembles (analyse, référence, etc.). L'introduction de la valeur joker * permet aussi d'extraire des séquences même si les items ne sont pas instanciés sur la totalité des dimensions d'analyse.

Par rapport à l'existant, cette définition est une généralisation des motifs séquentiels classiques et des motifs séquentiels multidimensionnels de [PHP+01]. Notre définition permet ainsi d'extraire des motifs qui ne peuvent pas l'être par les autres approches.

Toutefois, l'espace de recherche considéré est bien plus grand que :

Les motifs séquentiels "classiques" : Plusieurs dimensions sont considérées et une valeur joker * est introduite.

Les motifs séquentiels multidimensionnels de [PHP+01] : Les corrélations entre toutes les dimensions sont considérées et pas seulement des séquences multidimensionnelles *intra motifs multidimensionnels* (où une seule dimension varie au cours de la relation d'ordre).

Afin de proposer une approche d'extraction de tels motifs robuste, il est donc nécessaire :

- Soit de cibler certaines parties de l'espace de recherche en recherchant des types de connaissances précis,
- Soit de parcourir et élaguer efficacement l'espace de recherche afin d'extraire des connaissances non redondantes.

Dans les chapitres suivants, nous présentons et discutons ces deux solutions. Le chapitre 2 propose de cibler l'espace de recherche en découvrant des motifs séquentiels multidimensionnels à l'aide des items fréquents les plus spécifiques. Dans le chapitre 3, nous proposons de tout extraire. Pour cela, nous définissons les motifs séquentiels multidimensionnels clos qui permettent à la fois d'élaguer efficacement l'espace de recherche et d'extraire un ensemble non redondant de connaissance.

Chapitre 2

Extraction à Partir Des Items Les Plus Spécifiques

2.1 Introduction

Dans ce chapitre, nous définissons l'algorithme M^2SP (Mining Multidimensional Sequential Patterns). Avec cet algorithme, nous proposons d'extraire des motifs séquentiels multidimensionnels à partir des items fréquents maximalement spécifiques. Nous ciblons donc l'espace de recherche en découvrant des motifs séquentiels multidimensionnels composés d'items maximalement spécifiques.

L'algorithme se divise en deux étapes principales :

- La première étape consiste à extraire les items fréquents les plus spécifiques. Les items fréquents sont des séquences fréquentes composées d'un seul item au sein d'un unique itemset. Ils sont donc les éléments de base des séquences. Pour les extaire, nous utilisons une méthode de génération par niveau basée sur la paradigme *Apriori* [AS94].
- La deuxième étape consiste à extraire des motifs séquentiels multidimensionnels à partir de l'ensemble des items fréquents maximalement spécifiques. Pour cela, nous utilisons une approche de type *générer/élaguer* basée sur le paradigme *Apriori* [AS94]. Nous pouvons également réécrire la base de séquences de données multidimensionnelles en une base de séquences de données classiques afin d'appliquer des algorithmes d'extraction de motifs séquentiels classiques. La transformation des séquences de données multidimensionnelles s'appuie sur une numérotation des items multidimensionnels fréquents les plus spécifiques.

Ce chapitre s'organise de la façon suivante. Tout d'abord, nous définissons les items *maximalement spécifiques* et nous proposons un algorithme permettant leur extraction. Ensuite, nous montrons comment extraire des séquences multidimensionnelles à partir des items fréquents les plus spécifiques. Dans la section 2.4, nous définissons une méthode pour calculer le support d'une séquence. Nous étudions le comportement de M^2SP à l'aide d'expérimentations menées sur des jeux de données synthétiques et

réels dans la section 2.5. Enfin, nous discutons des perspectives associées à cette proposition dans la section 2.6.

Algorithme 1 : Algorithme Général M^2SP

Données : $DB, D_R, D_A, D_T, D_I, \sigma$

Résultat : L'ensemble des motifs séquentiels multidimensionnels construits à partir des items fréquents les plus spécifiques

début

Extraction des items fréquents maximalement spécifiques ;

Transformation de la base;

Extraction de séquences classiques;

retourner l'ensemble des motifs séquentiels multidimensionnels construits à partir des items fréquents les plus spécifiques;

fin

2.2 Extraction des Items les Plus Spécifiques

Afin de contourner le problème de la taille de l'espace de recherche, nous optons pour extraire des motifs séquentiels multidimensionnels à partir des items les plus spécifiques.

Définition 2.1 (Item Maximalement Spécifique)**.** Soit un seuil de support minimum σ, un item multidimensionnel fréquent e $(support(e) \geq \sigma)$ est dit maximalement spécifique s'il n'existe pas d'item e' tel que $e' \subset e$ et $support(e') \geq \sigma$.

Un item maximalement spécifique est un item *fréquent* tel qu'il n'existe pas d'item fréquent plus spécifique. De tels items représentent le niveau de granularité le plus fin des connaissances extraites.

Exemple 2.1. Considérons l'ensemble des items fréquent $\mathrm{I} = \{(1,2),(1,1),(3,*)(1,*),(*,1),(*,2)\}$, les items $(1,2)$, $(1,1)$ et $(3,*)$ sont maximalement spécifiques. Ce sont les seuls puisque $(1,2) \subseteq (1,*),(1,1) \subseteq (*,1)$ et $(1,2) \subseteq (*,2)$.

Les items multidimensionnels fréquents sont la base de l'extraction des motifs séquentiels multidimensionnels. Ils représentent les séquences fréquentes de taille 1 (appelés également 1-fréquents) puisqu'ils correspondent à des séquences composées d'un seul item au sein d'un seul itemset.

Dans certains cas, il est possible de découvrir tous les items fréquents (donc les plus spécifiques) en une seule passe sur la base de données comme dans le cas classique de l'extraction des items mono-attributs. Toutefois, selon la densité du jeu de données (cardinalité des dimensions d'analyse, nombre de combinaisons présentes, etc.), la gestion de tous les items potentiellement fréquents en une seule passe sur la base risque de ne pas tenir en mémoire. En effet, dès qu'une combinaison est présente, il est

nécessaire de la considérer dans un contexte « *une seule passe* ». De plus, une combinaison définie sur m dimensions d'analyse induit $2^m - 1$ items multidimensionnels potentiels à gérer. Par exemple, si le couple (1,2) défini sur deux dimensions d'analyse est présent dans la base de données, il faut alors considérer les items $(1, 2), (1, *)$ et $(*, 2)$, les items les plus spécifiques étant calculés une fois que tous les items fréquents sont extraits. Afin de limiter l'opération de calcul du support aux items dont la probabilité d'être fréquents est non-nulle, nous adoptons une méthode de génération par niveau basée sur la paradigme *Apriori* [AS94]. La recherche des items fréquents s'effectue ainsi au sein d'un treillis dont la bordure sera retenue pour constituer l'ensemble des items fréquents maximalement spécifiques. Ainsi, le premier niveau du treillis considère les items multidimensionnels pour lesquels une seule dimension d'analyse est spécifiée, les autres dimensions étant instanciées avec la valeur joker *. Les items multidimensionnels fréquents sont alors joints entre eux pour obtenir la liste des items candidats pour lesquels deux dimensions d'analyse sont spécifiées parmi lesquels on ne retient que les items fréquents. Cette procédure est itérée tant que de nouveaux items fréquents sont découverts.

La figure 2.1 illustre la recherche par niveau des items fréquents par l'intermédiaire du treillis des cuboïdes. La figure 2.2 propose un parcours arborescent de ce treillis. On note que cette exploration est très proche de l'exploration du treillis des cuboïdes dans le cadre des *iceberg cubes* [BR99, CCL03]. Dans ce contexte, il s'agit de ne considérer que les nœuds du treillis validant une condition définie par l'utilisateur (e.g. valeur de mesure supérieure à 5000). La contrainte d'iceberg, dans notre cas, est le support d'un item et non plus une contrainte sur la mesure des cellules d'un cube de données. L'amélioration de cette exploration dans le cadre de notre proposition s'effectuera en relation avec ces approches.

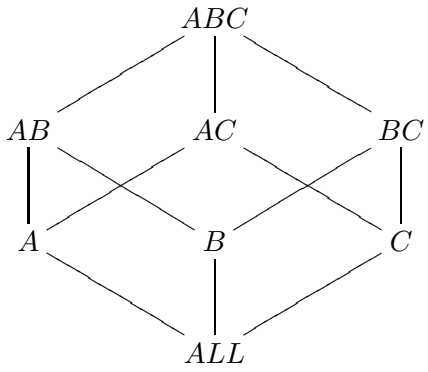


FIG. 2.1 – Le treillis des cuboïdes pour A, B et C

L'opération de *jointure* entre deux items fréquents suppose que les items partagent un nombre suffisant de valeurs de dimensions d'analyse (voir définition 2.3). On les dit alors *compatibles*.

Exemple 2.2. La figure 2.3 illustre l'approche de spécification par niveau pour les dimensions d'analyse A, B et C avec :

- $dom(A) = \{a, a'\}$
- $dom(B) = \{b, b'\}$
- $dom(C) = \{c\}$

Tout d'abord, les items où une seule dimension d'analyse est définie ($\neq *$) sont considérés au niveau 1. Ainsi, il s'agit de tester si les candidats $(a, *, *)$, $(a', *, *)$, ..., $(*, *, c)$ sont fréquents. Supposons que $(*, b', *)$ ne soit pas fréquent. Les items candidats de niveau 2 sont alors construits en combinant les items fréquents de niveau 1 (une seule dimension spécifiée). Ainsi, $(a, b, *)$, ..., $(*, b, c)$ sont potentiellement fréquents. Il faut donc vérifier leur fréquence. Dans notre exemple, seuls les items non soulignés sont fréquents. On construit alors les candidats de niveau 3. Par exemple, l'item candidat (a, b, c) est construit en combinant $(a, b, *)$ et $(a, *, c)$. La génération par niveau s'arrête au niveau 3 puisqu'à ce niveau toutes les dimensions sont instanciées. Finalement, les items maximalement spécifiques sont (a', b, c), $(a, b, *)$ et $(a, *, c)$.

Un élagage est possible avant même la vérification sur la base. Par exemple, si $(*, b, c)$ n'avait pas été fréquent, il aurait été impossible que (a', b, c) le soit, même si $(a', b, *)$ et $(a', *, c)$ l'étaient.

La figure 2.4 met en évidence le fait qu'ignorer une dimension en la mettant dans D_I n'est pas équivalent à instancier cette dimension par la valeur joker. En effet, les items multidimensionnels fréquents extraits ne sont pas les mêmes. Par exemple si $B \in D_I$, nous ne pouvons pas extraire l'item (a', b, c).

Définition 2.2 ($\bowtie$-compatibilité).
Soient deux items multidimensionnels $e_1 = (d_1, \cdots, d_n)$ et $e_2 = (d'_1, \cdots, d'_n)$ tels que $\forall i = 1, \ldots, n$, d_i et d'_i appartiennent à l'ensemble $dom(D_i) \cup \{*\}$. On dit que e_1 et e_2 sont $\bowtie$-compatibles si

- e_1 et e_2 sont distincts
- $\exists \Delta = \{D_{i_1}, \ldots, D_{i_{n-2}}\} \subset \{D_1, \ldots, D_n\}$ t.q. $d_{i_1} = d'_{i_1} \neq *$, $d_{i_2} = d'_{i_2} \neq *$, ... et $d_{i_{n-2}} = d'_{i_{n-2}} \neq *$
- Pour $\{D_{i_{n-1}}, D_{i_n}\} = \{D_1, \ldots, D_n\} \backslash \Delta$, on a $d_{i_{n-1}} = *$ et $d'_{i_{n-1}} \neq *$ et $d_{i_n} \neq *$ et $d'_{i_n} = *$

Pour être $\bowtie$-compatibles, deux items multidimensionnels définis sur n dimensions doivent donc partager $n - 2$ valeurs de dimension. Par exemple, $(Chicago, A, *)$ et $(*, A, rustine)$ sont deux items définis sur 3 dimensions d'analyse et partagent $3 - 2 = 1$ valeur (A sur la dimension Age). Ils sont donc $\bowtie$-compatibles. En revanche, les items $(Chicago, B, *)$ et $(NY, A, *)$ ne sont pas $\bowtie$-compatibles.

Définition 2.3 (Jointure). Soient deux items multidimensionnels $\bowtie$-compatibles $e_1 = (d_1, \ldots, d_n)$ et $e_2 = (d'_1, \ldots, d'_n)$. On définit $e_1 \bowtie e_2 = (v_1, \ldots, v_n)$ de la façon suivante :

- $v_i = d_i$ si $d_i = d'_i$
- $v_i = d_i$ si $d'_i = *$
- $v_i = d'_i$ si $d_i = *$

Soient deux ensembles d'items multidimensionnels de même taille n E et E', on note : $E \bowtie E' = \{e \bowtie e' \ t.q. \ (e, e') \in E \times E'$ et e et e' sont $\bowtie$-compatibles$\}$

Par exemple, la jointure des deux items $(Chicago, A, *)$ et $(*, A, rustine)$ est $(Chicago, A, rustine)$.

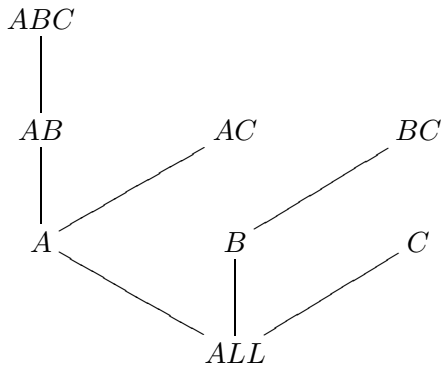


FIG. 2.2 – Arbre recouvrant du treillis des cuboïdes

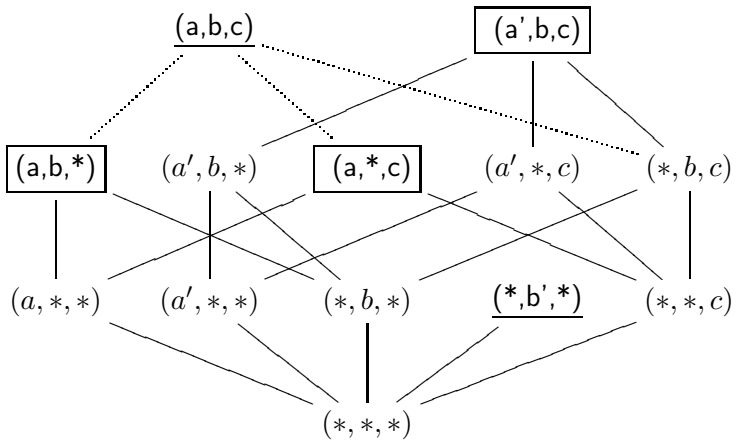


FIG. 2.3 – Les items multidimensionnels fréquents maximalement spécifiques avec $B \in D_A$

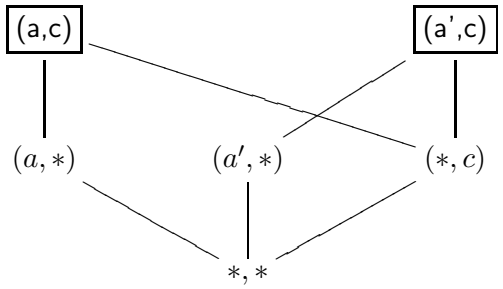


FIG. 2.4 – Les items multidimensionnels fréquents maximalement spécifiques avec $B \in D_I$

Notation On note F_1^i l'ensemble des 1-fréquents dont i dimensions sont spécifiées (différentes de $*$).

L'ensemble F_1^1 des 1-fréquents de taille 1 (les items fréquents ayant une seule valeur différente de *) est obtenu sans génération de candidats de taille 1. En effet, cet étape peut être fait en une seule passe sur la base sans génération de candidats comme pour les algorithmes d'extraction d'itemsets ou de séquences classiques.

L'ensemble F_1^2 des 1-fréquents de taille 2 est obtenu à l'aide de l'ensemble $Cand_1^2$ des candidats de taille 2. De manière générale, les candidats de taille i sont générés en considérant les 1-fréquents de taille $i-1$ (*cf.* algorithme 2) et permettent de découvrir les 1-fréquents de taille i :

$F_1^2 = \{f \in Cand_1^2,\ support(f) \geq \sigma\}$

$F_1^i = frequents(F_1^{i-1} \bowtie F_1^{i-1})$

Algorithme 2 : Extraction des items fréquents maximalement spécifiques

Données : $DB, D_R, D_A, D_T, D_I, \sigma$

Résultat : L'ensemble des items multidimensionnels fréquents maximalement spécifiques

début

$i \longleftarrow 1$;

$F_1 = F_1^1 \longleftarrow \emptyset$;

tant que $F_1^i \neq \emptyset \wedge i \leq m$ **faire**

$Cand_1^{i+1} \longleftarrow \emptyset$;

pour chaque *couple* $(e_1, e_2) \in F_1^{i^2}$ *tel que* e_1, e_2 $\bowtie$-*compatibles* **faire**

$Cand_1^{i+1} \longleftarrow Cand_1^{i+1} \cup \{e_1 \bowtie e_2\}$;

$F_1^{i+1} \longleftarrow \{f \in Cand_1^{i+1},\ support(f) \geq \sigma\}$;

$F_1 \longleftarrow F_1 \cup F_1^{i+1}$;

$i \longleftarrow i + 1$;

`/* on retourne les items les plus spécifiques */`

retourner ($\{e \in F_1$ t.q. e maximalement spécifique $\}$);

fin

2.3 Extraction des motifs séquentiels multidimensionnels

Les items multidimensionnels fréquents les plus spécifiques forment les unités de base des séquences que nous souhaitons extraire.

Les k-séquences candidates ($k \geq 2$) (séquences candidates contenant k items multidimensionnels) sont générées et testées afin de savoir si elles sont fréquentes. Cette opération est itérée tant que des k-candidats fréquents sont trouvés.

Pour extraire les motifs séquentiels multidimensionnels à partir de l'ensemble F_1^S des items maximalement spécifiques, nous pouvons également nous appuyer sur des algorithmes efficaces d'extraction de

B_1	1	(a_1, b_1, c_1)
	2	(a_1, b_2, c_2)
B_2	1	(a_1, b_1, c_1)
	2	(a_2, b_1, c_2)
B_3	1	(a_2, b_2, c_2)
	2	(a_1, b_1, c_2)

TAB. 2.1 – Base de séquences de données multidimensionnelles

motifs séquentiels classiques tels que PrefixSpan [PHMA+04] et Spade [Zak01]. Pour cela, il est nécessaire de transformer les données multidimensionnelles fouillées en un ensemble de séquences de données analysable par des algorithmes d'extraction de motifs séquentiels.

Nous transformons donc chaque item multidimensionnel e de F_1^S en un entier unique. On peut donc transformer la base de séquences de données multidimensionnelles en une base de séquences de données composées des entiers correspondants aux items fréquents les plus spécifiques. La recherche de motifs séquentiels classiques peut être ensuite réalisée avec n'importe quel algorithme d'extraction de motifs séquentiels. Nous choisissons l'algorithme Spade [Zak01] dans notre proposition car c'est l'un des algorithmes les plus efficaces.

Notons que si un bloc B_i ne supporte pas d'item maximalement spécifique, alors l'opération de réécriture de la base de séquences de données multidimensionnelles associera la séquence vide $\langle\rangle$ à ce bloc afin de ne pas introduire de biais dans le calcul du support des séquences (pour un calcul de support relatif).

Exemple 2.3 (D'un ensemble de séquences multidimensionnelles à un ensemble de séquences classiques)**.** Soit la base de séquences multidimensionnelles décrite par le tableau Tab. 2.1. Cette base représente 3 séquences de données définies sur 3 dimensions d'analyse. Nous considérons un seuil de support minimum σ fixé à 2. Dans un premier temps, nous extrayons les items multidimensionnels les plus spécifiques (les items en gras dans la figure 2.5). Ensuite, chaque item maximalement spécifique est associé à un unique entier. Le tableau Tab. 2.2 illustre la numérotation des items multidimensionnels les plus spécifiques. Par exemple, l'item (a_1, b_1, c_1) est associé à l'entier 1. A partir de cette numérotation, nous pouvons réécrire la base de séquences de données multidimensionnelles sous la forme de base de séquences de données classiques. Le tableau Tab. 2.3 montre la base de séquences de données classiques issue de la réécriture de la base de séquences de données multidimensionnelles (Tab. 2.1) à partir de la numérotation des items maximalement spécifiques.

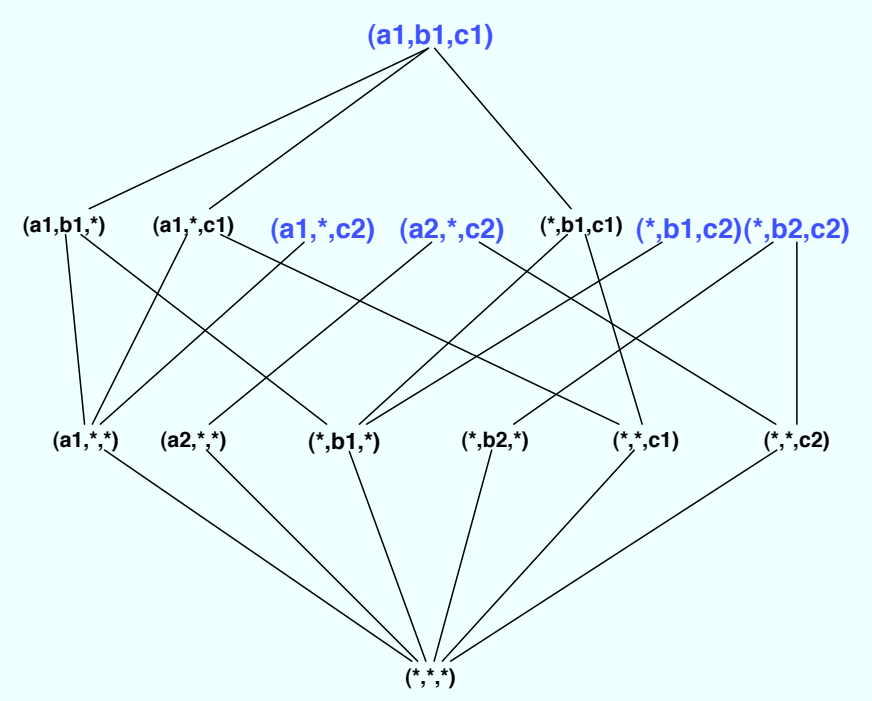


FIG. 2.5 – Les items maximalement spécifiques pour $\sigma = 2$

$$
\begin{array}{rcl}
(a_1, b_1, c_1) & \mapsto & 1 \\
(a_1, *, c_2) & \mapsto & 2 \\
(a_2, *, c_2) & \mapsto & 3 \\
(*, b_1, c_2) & \mapsto & 4 \\
(*, b_2, c_2) & \mapsto & 5
\end{array}
$$

TAB. 2.2 – Transformation des items multidimensionnels en items classiques

B_1	1	1
	2	$2, 5$
B_2	1	1
	2	$3, 4$
B_3	1	$3, 5$
	2	$2, 4$

TAB. 2.3 – Base de séquences de données « classiques » issue de la réécriture de Tab. 2.1

2.4 Calcul du support des séquences

Nous pouvons maintenant définir l'algorithme qui calcule le support d'une séquence ς au sein d'une base de données DB, suivant les dimensions de référence et d'analyse désirées.

Les dimensions de référence permettent d'énumérer tous les blocs de DB susceptibles de supporter ς. Cette énumération est indispensable pour calculer le ratio des blocs qui supportent ς, et donc pour définir si la séquence est fréquente ou non.

L'algorithme 3 vérifie pour chaque bloc de DB si la séquence est supportée ou non. Si la séquence est supportée, alors le support est incrémenté. L'algorithme retourne ensuite le ratio des blocs supportant ς.

L'algorithme 4 permet de vérifier si le bloc B_r supporte la séquence ς. Pour cela, cet algorithme cherche à instancier itemset par itemset en conjugant *récursivité* et *ancrage*. L'ancrage correspond à un tuple du bloc B_r d'où il est espéré que la séquence pourra être instanciée. Ce tuple correspond donc à une date à laquelle le premier item du premier itemset de la séquence est trouvé. À partir de cette cellule, seuls les tuples pertinents sont retenus, c'est-à-dire ceux qui partagent la même date. Si le sous-bloc résultant de l'ancrage supporte l'itemset alors on appelle la fonction sur les autres itemsets de ς. Cet appel est effectué en réduisant l'espace de recherche aux seuls tuples dont la date est supérieure à la date de l'ancrage précédent, puisque l'on passe à l'itemset suivant, donc à une date ultérieure. Si l'ancrage échoue, on continue la recherche du premier itemset en tentant d'autres ancrages. L'appel récursif s'arrête dès que la séquence placée en paramètre d'entrée est vide. Une telle propriété signifie en effet que tous les itemsets de la séquence ont été trouvés. On retourne donc la valeur $vrai$. La valeur $faux$ est retournée si aucun ancrage n'a réussi et si tout le bloc a été parcouru sans succès.

Algorithme 3 : Calcul du support d'une séquence (supportcount)

Données : ς, DB, D_R, D_T
Résultat : $support(\varsigma)$
début
  Entier $support \longleftarrow 0$
  Booleen $seqSupportée$
  $B_{DB,D_R} \longleftarrow$ {blocs de DB par rapport à D_R}
  pour chaque $B_r \in B_{DB,D_R}$ **faire**
    $seqSupportée \longleftarrow supportBloc(\varsigma, B_r)$
    si $seqSupportée$ **alors**
      $support \longleftarrow support + 1$
  retourner $\left(\frac{support}{|B_{DB,D_R}|}\right)$
fin

Algorithme 4 : Vérification si une séquence est supportée par un bloc donné (SupportBloc)

Données : ς, B_r, D_T

Résultat : Vrai si B_r supporte ς, Faux sinon.

début

```
/* initialisation                                                    */
booleen ItemSetTrouvé ⟵ faux
sequence ⟵ ς
itemset ⟵ sequence.first()
item ⟵ itemset.first()
/* condition d'arrêt de la recursivité                               */
si ς = ∅ alors
    retourner (vrai)
/* parcours du bloc                                                  */
tant que tuple ⟵ B_r.next ≠ ∅ faire
    si supporte(tuple, item) alors
        itemSuivant ⟵ itemset.second()
        si itemSuivant = ∅ alors
            itemsetTrouvé ⟵ vrai
        /* Recherche de tous les items de l'itemset                  */
        sinon
            /* On ancre par rapport à l'item (D_T)                   */
            B' ⟵ σ_{D_T=tuple.D_T}(B_r)
            tant que tuple' ⟵ B'.next() ≠ ∅ ∧ itemsetTrouvé = faux faire
                si supporte(tuple', itemSuivant) alors
                    itemSuivant ⟵ itemset.next()
                    si itemSuivant = ∅ alors
                        itemsetTrouvé ⟵ vrai
        si itemsetTrouvé = vrai alors
            /* recherche des autres itemset                          */
            retourner (supportBloc(sequence.tail(), σ_{D_T>tuple.date}(B_r)))
        sinon
            itemset ⟵ sequence.first()
            /* on ne veut plus revoir les tuple de B'                */
            B_r ⟵ σ_{date>cell.D_T}(B_r)
/* pas trouvé                                                        */
retourner (faux)
```

fin

Complexité

Afin de faciliter l'étude de complexité des algorithmes, nous posons les notations suivantes :

- n_{B_r} est le nombre de tuples du bloc B_r
- $m = |D_A|$ est le nombre de dimensions des items multidimensionnels.

supportBloc (algorithme 4)

- Le bloc B_r étant ordonné par rapport à la dimension D_t, l'opération d'ancrage est réalisable en $O(\log n_{B_r})$. En effet, il suffit de réaliser une recherche à l'aide d'un parcours dichotomique pour trouver tous les tuples respectant une certaine condition sur la date.
- Vérifier si une cellule supporte un item est réalisable en $O(m)$. Il suffit de comparer les m dimensions de l'item avec celles du tuple.
- Dans le pire des cas, la complexité de l'algorithme est de $O(n_{B_r} \times m \times \log n_{B_r})$.

supportcount (algorithme 3)

On appelle la fonction précédente pour tous les l blocs B_r de B_{DB,D_R}. Soit $n_{max} = \max n_{B_i}$. La complexité dans le pire des cas est donc $O(l \times n_{max} \times m \times \log n_{max})$.

2.5 Expérimentations

Dans cette section, nous rapportons les expérimentations effectuées sur des données synthétiques et réelles.

Nous avons effectué des expérimentations sur des jeux de données synthétiques afin d'étudier le comportement de M^2SP. Ces expérimentations visent à montrer le passage à l'échelle de cette proposition. Nous étudions ainsi le comportement (temps d'exécution) de M^2SP en fonction de plusieurs paramètres tels que le nombre de dimensions d'analyse, le seuil de support minimum, la taille de la base de données et la cardinalité moyenne des dimensions d'analyse.

Par défaut, le jeux de données synthétiques utilisé est composé de 5 dimensions d'analyse de cardinalité moyenne égale à 350, 50000 blocs différents (environ un million de n-uplets).

La figure 2.6(a) décrit le temps d'exécution de M^2SP en fonction du nombre de dimensions d'analyse considérées pour différents seuils de support minimum. Augmenter le nombre de dimensions d'analyse revient à augmenter « en largeur » la taille du jeu de données considéré. En effet, même si le nombre de n-uplets reste le même, les n-uplet sont décrits sur plus dimensions. Le parcours de l'ensemble de données est donc plus long car il faut considérer plus de dimensions pour chaque n-uplet. De plus, lorsque le nombre de dimensions d'analyse augmente, le nombre d'items multidimensionnels fréquents augmente également. Par exemple, supposons l'item (a, b) fréquent sur deux dimensions d'analyse A et B, si on ajoute une dimension d'analyse C, alors dans le pire des cas (aucune valeur fréquente sur C), nous obtiendrons l'item $(a, b, *)$, sinon des items de la forme (a, b, c) et (a, b, c') peuvent apparaître. Le graphe de la figure 2.6(b) illustre l'augmentation du nombre de motifs extraits en fonction du nombre de dimensions d'analyse.

L'augmentation du nombre de motifs extraits est principalement due à l'augmentation du nombre d'items maximalement spécifiques fréquents. En fonction de ces différents points, nous pouvons supposer que le temps d'exécution va augmenter en fonction du nombre de dimensions d'analyse considérées. La courbe de la figure Fig 2.6(a) confirme ceci. Le temps d'exécution de M^2SP augmente lorsque le nombre de dimensions d'analyse augmente. Plus le nombre de dimensions d'analyse est important, plus la pente de la courbe augmente. La pente de la courbe est également plus forte si le seuil de support considéré est moins contraignant. Ceci s'explique par le fait que M^2SP extrait plus de motifs quand le seuil de support est faible.

Nous ne pouvons pas conclure au passage à l'échelle de M^2SP en fonction du nombre de dimensions d'analyse considérées puisque le temps d'exécution de M^2SP semble augmenter de manière exponentielle avec le nombre de dimensions d'analyse. Ainsi, au delà de 20 dimensions d'analyse, l'extraction ne semble pas envisageable pour des supports faibles. Toutefois, nous pouvons modérer le non passage à l'échelle de M^2SP en fonction du nombre de dimensions d'analyse car en pratique, nous ne connaissons pas d'applications nécessitant plus de 20 dimensions d'analyse. De plus, l'interprétation des motifs extraits est très difficile quand le nombre de dimensions d'analyse est très important. Il est également possible de « réduire » le nombre de dimensions d'analyse en découvrant des dépendances fonctionnelles entre certaines dimensions.

La figure 2.6(c) décrit le temps d'exécution de M^2SP en fonction de la taille de la base de données pour différentes valeurs de seuils de support (0.5, 0.25 et 0.1) alors que la figure 2.6(d) décrit également le temps d'exécution de M^2SP en fonction de la taille de la base de données mais pour un nombre différent de dimensions d'analyse (2,5 et 10). La taille de la base de données est indiquée en nombre de blocs. Dans cette expérimentation, la taille de la base de données examinée varie de 1000 blocs à 500000 blocs. 500000 blocs correspondent à environ 10 million de n-uplets. Dans les deux graphes, le temps d'exécution de M^2SP augmente proportionnellement à la taille de la base de données. La pente de chaque courbe dépend du nombre de dimensions d'analyse considérées (Fig. 2.6(d)) ou du seuil de support minimum fixé (Fig 2.6(c)).

Nous pouvons conclure au passage à l'échelle de M^2SP en fonction de la taille de la base de données. En effet, le comportement de notre approche évolue proportionnellement à la taille de la base de données.

La figure 2.6(e) rapporte le temps d'exécution de M^2SP en fonction du seuil de support minimum ($minsupp$) pour différents choix du nombre de dimensions d'analyse (5, 10 et 15). Rappelons que dans les problèmes d'extraction de motifs, le nombre de motifs extraits devient très (trop) important quand le seuil de support minimum devient faible, ce qui entraîne des temps d'extraction relativement importants pour ces seuils. M^2SP ne déroge pas à la règle. En effet, le temps d'exécution augmente quand le seuil de support minimum diminue. Ceci s'explique principalement par le fait que le nombre de combinaison à prendre en compte devient plus important. La pente de la courbe s'accentue quand le seuil de support minimum devient plus faible. A des supports très faibles (0.1), on assiste à une explosion combinatoire.

Le temps d'exécution devient très important du fait du nombre de combinaisons à prendre en compte. Plus le nombre de dimensions d'analyse est important, plus l'explosion combinatoire se fait ressentir.

La figure 2.6(e) illustre un des problèmes majeurs inhérents à l'extraction de motifs : le choix de la bonne valeur de seuil de support minimum. En effet, si le seuil considéré est trop important, très peu de motifs sont extraits. De plus, les motifs extraits sont généralement triviaux et sont donc d'aucune utilité pour l'utilisateur. Si le seuil de support fixé est trop faible, le nombre de motifs extraits est trop important, ce qui rend impossible l'utilisation des ces motifs par un utilisateur humain. Dans notre contexte multidimensionnel, le problème reste identique.

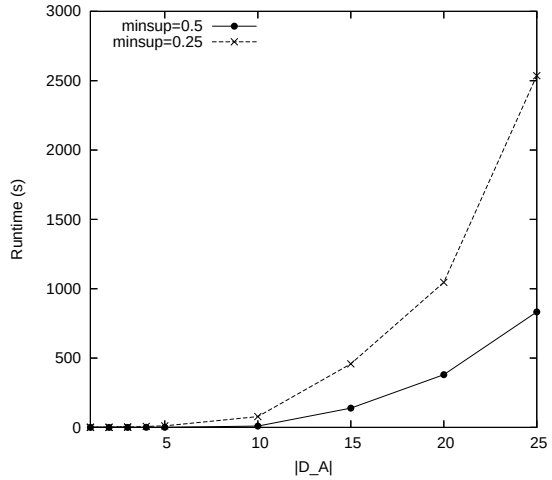
La figure 2.6(f) décrit le temps d'exécution de M^2SP en fonction de la cardinalité moyenne des dimensions d'analyse pour différents choix du nombre de dimensions d'analyse. Les expérimentations rapportées par cette figure consistent à faire varier la cardinalité moyenne des dimensions d'analyse pour un ensemble de n-uplets fixé. Ainsi, augmenter la cardinalité moyenne des dimensions d'analyse revient à « perdre » des motifs fréquents. En effet, si la cardinalité des dimensions d'analyse augmente, le nombre de combinaisons possibles à prendre en compte augmente également. Or comme le nombre de n-uplets reste identique, la probabilité de découvrir des motifs diminue. Le temps d'exécution de M^2SP augmente proportionnellement à la cardinalité des dimensions d'analyse. Ceci est dû au coût de l'extraction des items contenant une valeur différente de *. En effet, nous utilisons des tables de hachages et le nombre d'entrées devient très important. Nous pouvons toutefois conclure au passage à l'échelle de notre approche en fonction de ce paramètre.

Ces expérimentations menées sur des données synthétiques montrent l'intérêt de notre proposition. En effet, M^2SP permet d'extraire des motifs séquentiels multidimensionnels à partir des items les plus spécifiques avec des seuils de support assez faibles, un nombre de dimensions d'analyse respectable, des bases de données de taille très importante, etc.

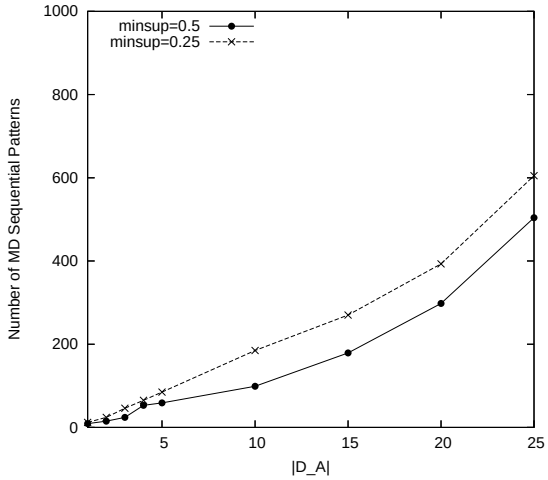
La figure 2.7(a) décrit le temps d'exécution de M^2SP en fonction du seuil de support minimum. Cette figure montre l'efficacité de notre approche sur des données réelles, les données EDF décrites dans l'annexe A. La figure 2.7(b) décrit le nombre de motifs fréquents extraits par M^2SP. Le nombre de motifs extraits augmente lorsque le seuil de support minimum diminue. Le temps d'exécution traduit ce comportement. Il augmente également lorsque le support diminue.

2.6 Discussion

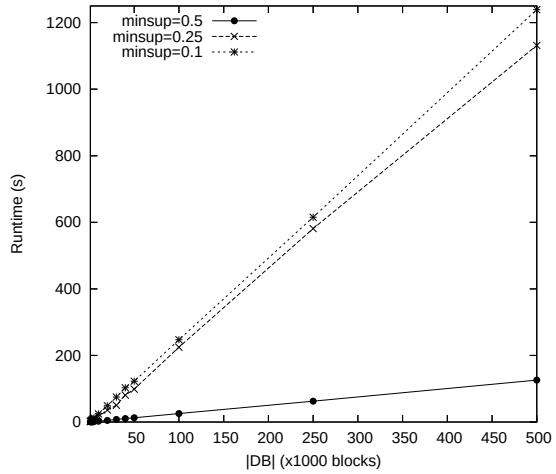
L'algorithme M^2SP permet d'extraire des motifs séquentiels multidimensionnels à partir des items fréquents les plus spécifiques. L'algorithme est divisé en deux étapes. Tout d'abord les items fréquents les plus spécifiques sont extraits à l'aide du paradigme *g*énérer/élaguer. Ensuite, les motifs séquentiels multidimensionnels sont extraits à l'aide de l'ensemble des items fréquents les plus spécifiques. Cet algorithme propose de cibler l'espace de recherche trop important. Les expérimentations montrent le passage à l'échelle de cette proposition en fonction de nombreux paramètres.



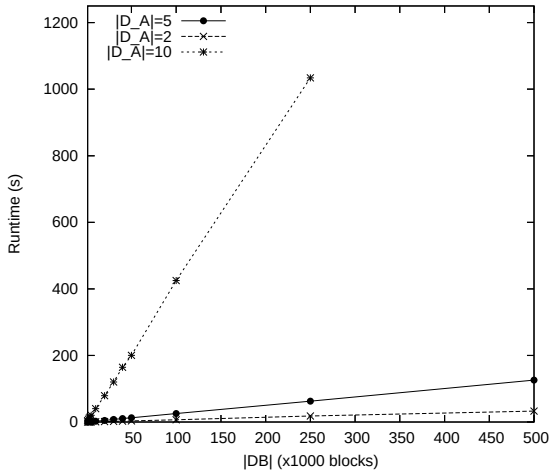
(a) Temps d'exécution en fonction du nombre de dimensions d'analyse



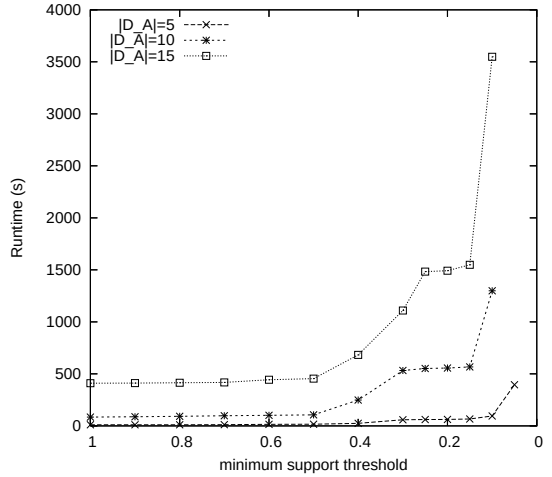
(b) Nombre de motifs extraits en fonction du nombre de dimensions d'analyse



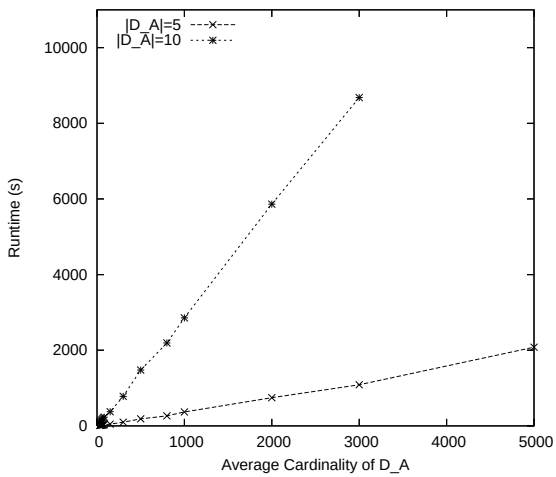
(c) Temps d'exécution en fonction de la taille de la base de données (différents $minsup$)



(d) Temps d'exécution en fonction de la taille de la base de données (différents $|D_A|$)

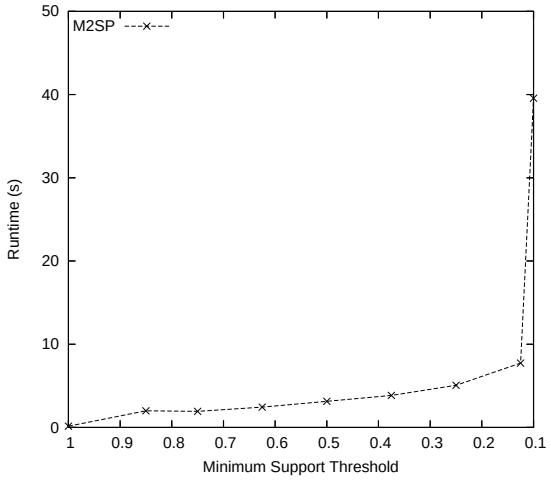


(e) Temps d'exécution en fonction du seuil de support minimum

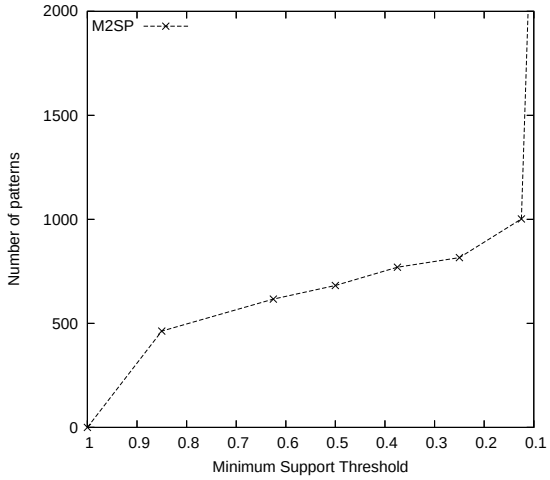


(f) Temps d'exécution en fonction de la cardinalité moyenne des dimensions d'analyse

FIG. 2.6 – Expérimentations menées sur des jeux de données synthétiques



(a) Temps d'exécution en fonction du seuil de support minimum



(b) Nombre de motifs fréquents en fonction du seuil de support minimum

FIG. 2.7 – Expérimentations menées sur des jeux de données synthétiques

Cependant, le passage à l'échelle de M^2SP par rapport au nombre de dimensions d'analyse pose problème. Quand le nombre de dimensions d'analyse devient trop important (20), le temps d'exécution de M^2SP augmente considérablement. Toutefois, prendre en compte autant de dimensions d'analyse en pratique est rare car il est très difficile d'interpréter les motifs extraits. Il peut s'avérer judicieux de réduire le nombre de dimensions d'analyse de manière automatique afin de ne pas examiner inutilement une dimension. En effet, si les valeurs d'une dimension sont induites par des valeurs d'une autre dimension, il n'est pas utile de considérer cette dimension.Nous pouvons nous s'orienter vers les travaux relatifs à la recherche de dépendances fonctionnelles pour réduire le nombre de dimensions.

Notre proposition permet d'extraire les motifs séquentiels multidimensionnels à partir des items fréquents les plus spécifiques. Ainsi, tous les motifs séquentiels multidimensionnels ne sont pas découverts. M^2SP ne permet pas d'extraire des séquences composées d'items plus généraux. Il serait intéressant de continuer à cibler l'espace de recherche mais à partir d'autres caractéristiques. Par exemple, nous pourrions extraire des séquences multidimensionnelles à partir des items fréquents clos (item fréquent tel qu'il n'existe pas d'item plus spécifique de même support).

Les données réelles sont souvent multidimensionnelles. Elles sont également agrégées selon plusieurs niveaux de hiérarchie. Il est donc également nécessaire de prendre en compte ces hiérarchies dans l'extraction des motifs séquentiels multidimensionnels. Nous abordons ce problème dans la partie II.

Chapitre 3

Extraction de Motifs Séquentiels Multidimensionnels Clos

3.1 Introduction

Contrairement au chapitre précédent où nous proposons de cibler l'espace de recherche en extrayant les motifs séquentiels multidimensionnels composés des items maximalement spécifiques, nous souhaitons extraire, dans ce chapitre, l'ensemble complet des motifs séquentiels multidimensionnels. Il est donc nécessaire de proposer des méthodes efficaces de parcours et d'élagage de l'espace de recherche. De plus, parmi l'ensemble complet des motifs séquentiels multidimensionnels, il peut y avoir énormément de redondances. Il est donc nécessaire de présenter à l'utilisateur un ensemble *complet* et *non redondant* de motifs séquentiels multidimensionnels.

Nous souhaitons donc extraire un ensemble non redondant de connaissances tout en parcourant efficacement l'espace de recherche. Dans le contexte classique d'extraction de motifs (séquentiels ou non), la recherche de motifs clos permet de respecter ces deux contraintes. Les motifs clos sont des motifs tels qu'il n'existe pas de *super motifs* de même support. Ils permettent d'introduire des propriétés supplémentaires d'élagage de l'espace de recherche et offrent une représentation *condensée* sans perte d'information des connaissances extraites.

L'organisation de ce chapitre est la suivante. Tout d'abord, nous étudions les motifs clos dans le cadre de l'extraction d'itemsets ou de motifs séquentiels dans la section 3.2. Dans la section 3.3, nous définissons les motifs séquentiels multidimensionnels clos ainsi qu'un cadre formel pour leur extraction. Nous proposons également deux algorithmes d'extraction de motifs séquentiels multidimensionnels clos. Des expérimentations menées à la fois sur des données synthétiques et réelles montrent l'intérêt de l'utilisation d'une telle représentation (taille de l'ensemble non-redondant et temps d'extraction) dans la section 3.5. Enfin, nous discutons des perspectives associées à cette proposition dans la section 3.6.

3.2 Panorama des travaux existants

Défini par [PBTL99], un *motif clos ou fermé* est un motif qui n'a pas le même support que tous ses super-motifs. Les motifs (séquentiel) clos permettent de représenter les connaissances extraites de manière compacte sans perte d'information et sont généralement associés à des propriétés qui permettent de réduire sensiblement l'espace de recherche à l'aide d'opérations d'élagage autres que l'élagage élémentaire des motifs non fréquents basé sur l'antimonotonie du support.

Définition 3.1 (Motif Clos)**.** Un motif α est *clos* s'il n'existe pas de motif β tel que $\alpha \preceq \beta$ et $support(\alpha) = support(\beta)$.

Cette définition peut aussi s'écrire de la façon suivante : étant donné un opérateur de fermeture γ sur l'ensemble des motifs partiellement ordonné par l'inclusion ($\preceq$), un motif I est clos si et seulement si $I = \gamma(I)$.

Depuis [PBTL99], de nombreux travaux ont été effectués sur l'extraction d'itemsets clos [EHZ05, PHM00, ZH02, WK04, UAUA04, MFT06, PCT+03, YHN06].

Les motifs (séquentiels) clos offrent à la fois une représentation condensée des connaissances issues de la base de données sans perte d'information et des propriétés efficaces d'élagage de l'espace de recherche. Ainsi, à partir des motifs clos, le support de n'importe quelle séquence fréquente peut être inférée.

Le support des séquences fréquentes est directement corrélé au support des séquences fréquentes closes. Toutes les séquences fréquentes sont illustrées dans la figure 3.1 où les séquences closes sont cerclées.

Par exemple, soit l'ensemble des séquences closes $C = \{ \langle a\rangle_3, \langle a, b, c\rangle_2\}$ où $\langle a\rangle_3$ signifie que la séquence $\langle a\rangle$ a un support de 3. A partir de l'ensemble C, on peut déduire le support des séquences $\langle a, c\rangle$ et $\langle a, b\rangle$. Ces séquences ont un support égal à 2. En effet, elles sont incluses dans la séquence close $\langle a, b, c\rangle_2$, leur support est donc supérieur ou égal à 2. Le fait que la séquence $\langle a\rangle_3$ soit close signifie qu'il n'existe pas de super-séquence de $\langle a\rangle$ ayant un support égal à 3. Le support des séquences $\langle a, c\rangle$ et $\langle a, b\rangle$ est donc strictement inférieur à 3. Ainsi, le support des séquences $\langle a, c\rangle$ et $\langle a, b\rangle$ est égal à 2.

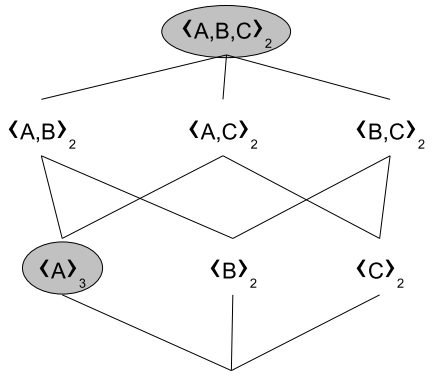


FIG. 3.1 – Recherche des séquences fréquentes à partir des closes.

Il faut noter que même s'il existe de nombreux travaux pour l'extraction d'itemsets clos [PBTL99, PHM00, ZH02, WK04, UAUA04, EHZ05, MFT06, PCT+03, YHN06], il n'y a, à notre connaissance, que deux propositions pour les motifs séquentiels clos : BIDE [WHL07] et CloSpan [YHA03].

CloSpan [YHA03] extrait tout d'abord l'ensemble des séquences closes candidates à partir de l'ensemble des séquences fréquentes. Puis, dans une seconde étape, CloSpan élimine les séquences candidates non closes. L'approche adoptée est une approche « *pattern-growth* ». La base de données est décomposée afin d'éviter les calculs inutiles [Pei02]. Ainsi, une base projetée (notée $DB|_\alpha$) est obtenue en fonction de la séquence préfixe examinée (α) afin de réduire la base projetée aux seules données permettant de construire des séquences de préfixe α. Par exemple, étant donnée la séquence de données $S = \langle (abcd)ea(bc)(ac)\rangle$ si $\alpha = \langle (ab)a\rangle$ alors $S|_\alpha = \langle (bc)(ac)\rangle$. Dès qu'une sous-séquence ou une super-séquence de l'espace de recherche partage la même base projetée que la séquence préfixe courante, il n'est pas nécessaire d'examiner toute la base de données lors de l'extension de la séquence préfixe. Les figures 3.2 et 3.3 illustrent comment deux arborescences de l'espace de recherche peuvent être réunies dès lors qu'elles partagent la même base projetée.

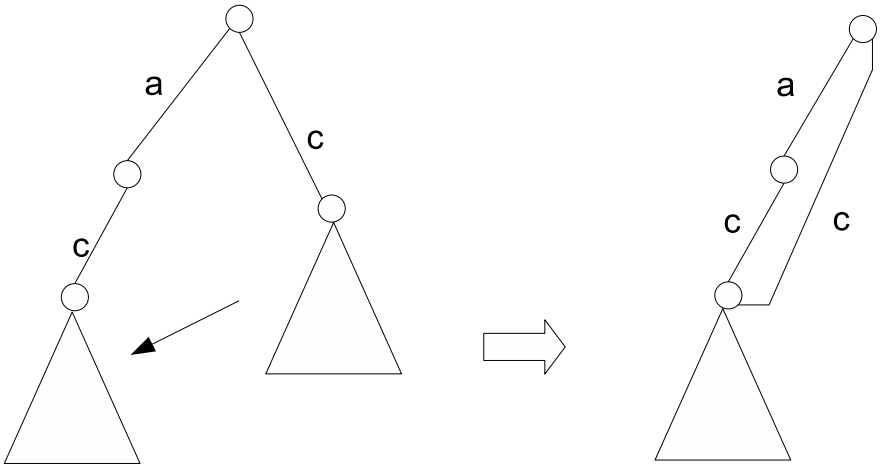


FIG. 3.2 – CloSpan : Backward Sub Pattern

BIDE [WHL07] étend l'approche précédente en proposant une approche sans gestion des candidats. Les séquences closes sont vérifiées à la volée au cours du processus d'extraction contrairement à CloSpan qui effectue cette vérification, quadratique en la taille de l'ensemble des candidats, à la fin de l'extraction des séquences fréquentes. Pour déterminer, si une séquence est close, l'algorithme BIDE vérifie s'il n'existe pas d'item qui puisse s'intercaler dans la séquence tout en conservant le support de celle-ci.

Soulignons le fait que ces deux approches ne proposent pas seulement une représentation condensée des connaissances extraites. En effet, BIDE et CloSpan introduisent des propriétés supplémentaires d'élagage de l'espace de recherche, ce qui leur permet d'obtenir de meilleures performances aussi bien en temps d'exécution qu'en espace mémoire que des approches classiques d'extraction de motifs séquentiels.

Nous pouvons également citer les travaux de [SBI06] qui abordent le problème des motifs séquentiels clos dans un contexte multidimensionnel en proposant une représentation condensée des motifs définis

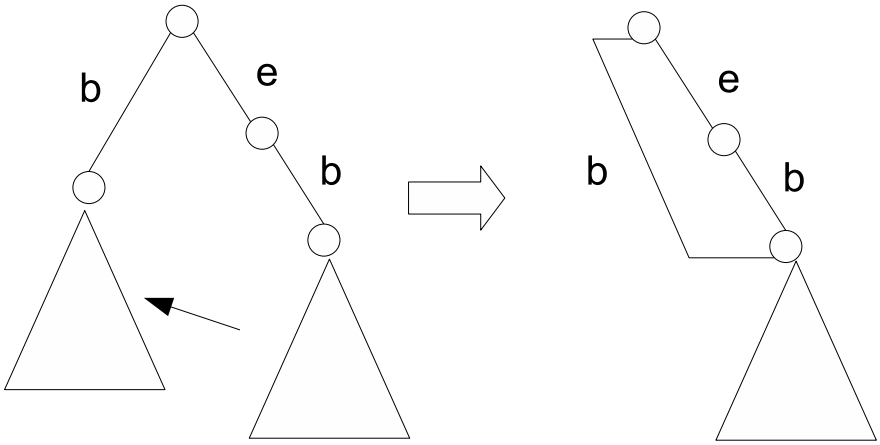


FIG. 3.3 – CloSpan : Backward Super Pattern

dans [PHP+01]. Dans ce cas, il s'agit de séquences définies sur une seule dimension (*e.g.* product) identifiée par un « motif » multidimensionnel. Il est alors toujours impossible de réaliser des combinaisons des dimensions au cours du temps au sein même de la séquence.

Ainsi, à notre connaissance, il n'existe aucune proposition offrant un cadre théorique pour une représentation condensée des motifs séquentiels multidimensionnels. Dans la section 3.3, nous définissons les motifs séquentiels multidimensionnels clos.

3.3 Motifs Séquentiels Multidimensionnel Clos

Dans cette section, nous montrons que la définition classique des motifs séquentiels clos ne peut pas s'appliquer directement dans un contexte multidimensionnel. En effet, une séquence multidimensionnelle peut être incluse dans une autre si elle contient moins d'items ou si elle contient des items plus spécifiques. Nous définissons une nouvelle inclusion pour que la définition des motifs clos s'adapte à notre contexte multidimensionnel.

3.3.1 Les Limites de l'inclusion

La définition classique des motifs clos trouve ses limites dans le contexte multidimensionnel dès que la valeur joker est adoptée. Par exemple, si nous supposons que la séquence $s = \langle\{(a_1, b_1), (*, b_2)\}\rangle_2$ est la seule séquence close, ceci implique par définition des motifs clos, que les sous séquences de s telles que $\langle\{(a_1, b_1), (a_1, b_2)\}\rangle$ et $\langle\{(a_1, b_1), (a_2, b_2)\}\rangle$ ont également un support égal à 2. Néanmoins, il est possible d'obtenir pour chacune de ces deux séquences un support égal à 1, car l'inclusion, ici, est associée au caractère * qui est plus général que n'importe quelle valeur considérée. Le tableau suivant illustre une base de données où $s = \langle\{(a_1, b_1), (*, b_2)\}\rangle_2$ est close, mais où les séquences $\langle\{(a_1, b_1), (a_1, b_2)\}\rangle$ et $\langle\{(a_1, b_1), (a_2, b_2)\}\rangle$ ont un support égal à 1.

s_1	$\langle\{(a_1,b_1),(a_1,b_2)\}\rangle$
s_2	$\langle\{(a_1,b_1),(a_2,b_2)\}\rangle$

Cet exemple montre les limites d'une utilisation directe de la définition des motifs clos et notamment de l'inclusion dans un contexte multidimensionnel. En effet, dès lors que la valeur joker est considérée, l'inclusion d'une séquence dans une autre peut apparaître de deux façons différentes :

- Une séquence α peut être incluse dans une séquence β si β contient tous les itemsets (et tous les items) de α. Ainsi, la séquence β est plus longue que la séquence α.
  Par exemple $\langle\{(a_1,b_2),(a_1,b_3)\}\{(a_2,b_2)\}\rangle \prec \langle\{(a_1,b_2),(a_1,b_3)\}\{(a_1,b_2)(a_2,b_2)\}\{(a_1,b_1)\}\rangle$.
- Une séquence α peut aussi être incluse dans une séquence β si β contient des items *plus généraux*. En effet, par définition, la valeur joker est équivalent à ALL. Ainsi, un item qui contient des * est plus général qu'un autre qui n'en contient pas. Par exemple, $\langle\{(a_1,b_2),(a_1,b_3)\}\{(a_2,b_2)\}\rangle \prec \langle\{(*,b_2),(*,b_3)\}\{(a_2,*)\}\rangle$.

3.3.2 Une nouvelle inclusion pour les motifs séquentiels multidimensionnels clos

Dans un contexte multidimensionnel, une séquence peut être plus spécifique qu'une autre si elle contient plus d'items (séquence plus longue), et/ou si elle contient des items plus spécifiques (moins de valeurs *).

Définition 3.2 (Spécialisation/Généralisation). Un motif séquentiel multidimensionnel $\alpha = \langle a_1, a_2, \ldots, a_l\rangle$ est plus général que $\beta = \langle b_1, b_2, \ldots, b_{l'}\rangle$ $(l \leq l')$ (et β plus spécifique que α) s'il existe des entiers $1 \leq j_1 \leq j_2 \leq j_l \leq l'$ tels que $b_{j_1} \subseteq a_1, b_{j_2} \subset a_2, \ldots, b_{j_l} \subseteq a_l$.

Si β est plus spécifique que α, nous notons $\alpha \prec_S \beta$ où $\prec_S$ représente la relation de spécialisation.

Exemple 3.1. Soient $s_1 = \langle\{(a_1,b_1,c_1),(a_2,*,c_1)\}\{(*,b_2,c_2)\}\rangle$, $s_2 = \langle\{(a_1,*,*),(a_2,*,c_1)\}\{(*,b_2,c_2)\}\rangle$ et $s_3 = \langle\{(a_1,b_1,c_1)\}\{(*,b_2,c_2)\}\rangle$ trois séquences multidimensionnelles. On a :

- $s_2 \prec_S s_1$,
- $s_3 \prec_S s_1$.

A partir de cette définition, nous pouvons maintenant proposer une définition d'une séquence multidimensionnelle close.

Définition 3.3 (Séquence Multidimensionnelle Close). Une séquence multidimensionnelle α est close s'il n'existe pas de séquence β telle que $\alpha \prec_S \beta$ et $support(\alpha) = support(\beta)$.

La définition des motifs séquentiels multidimensionnels clos découle naturellement de la définition précédente.

Définition 3.4 (Motif Séquentiel Multidimensionnel Clos).
Etant donné un seuil minimum de support σ, une séquence multidimensionnelle s est un motif séquentiel multidimensionnel clos si s est close et $support(s) \geq \sigma$.

Exemple 3.2 (Motifs Séquentiels Multidimensionnels Clos et Inférence).
Considérons la base de séquences de données multidimensionnelles DB représentée par le tableau Tab. 3.1. Cette base contient 3 blocs différents (1,2 et 3). Chacun de ces blocs identifie la séquence de données qui lui est propre. Par exemple, le bloc 1 identifie la séquence de données $\langle\{(a_1,b_1),(a_1,b_2)\}\{(a_2,b_2)\}$-$\{(a_1,b_3)\}\{(a_1,b_2)(a_2,b_2)\}\rangle$.

Etant donné un seuil minimum de support égal à 2, l'ensemble des motifs séquentiels multidimensionnels clos sont représentés dans le tableau Tab. 3.2. La première partie du tableau décrit les motifs clos dont le support est égal à 3 alors que dans la seconde partie du tableau, les clos de support 2 sont énumérés.

Les motifs séquentiels multidimensionnels non clos et leur support peuvent être inférés à partir des motifs clos. Par exemple, le support des motifs $\langle(*,b_2)\rangle$ et $\langle(a_2,*)\rangle$ est égal à 3.

Deux niveaux de connaissances sont inférés à partir des motifs séquentiels multidimensionnels clos :

1. les sous-séquences contenant moins d'items,
2. les sous-séquences (par rapport à $\prec_S$) contenant des items plus généraux (plus de *).

Ces deux niveaux permettent d'inférer des connaissances plus générales.

1	$\langle\{(a_1,b_1),(a_1,b_2)\}\{(a_2,b_2)\}\{(a_1,b_3)\}\{(a_1,b_2)(a_2,b_2)\}\rangle$
2	$\langle\{(a_1,b_2),(a_2,b_1)\}\{(a_3,b_2)\}\{(a_2,b_1)\}\{(a_2,b_1)\}\rangle$
3	$\langle\{(a_4,b_4)\}\{(a_2,b_1)\}\{(a_1,b_1)(a_2,b_2))\}\rangle$

TAB. 3.1 – Base de Données Exemple

La problématique générale de l'extraction de motifs séquentiels multidimensionnels clos est donc la suivante : *Etant donné un seuil de support minimum fixé* a priori σ*, l'objectif de la recherche de motifs séquentiels multidimensionnels clos est d'extraire toutes les séquences multidimensionnelles closes dont le support est supérieur à* σ.

La résolution de ce problème dans un contexte multidimensionnel pose de nombreuses difficultés. Nous allons détailler dans la section suivante les problèmes ainsi que les solutions proposées.

3.4 CMSP : Extraction de motifs séquentiels multidimensionnels clos

Dans cette section, nous définissons des algorithmes d'extraction de motifs séquentiels multidimensionnels clos $CMSP$ (Closed Multidimensional Sequential Patterns). Nous présentons successivement : (1) l'adaptation du paradigme "*pattern growth*" dans un contexte multidimensionnel ; (2) $CMSP\_Cand$, un algorithme d'extraction de motifs séquentiels multidimensionnels clos qui est une adaption de CloSpan ; et enfin (3) $CMSP\_Free$, un algorithme qui ne gère aucun ensemble de candidats.

3.4.1 Approche “pattern growth" et ordre dans les itemsets

Les méthodes basées sur le paradigme *générer/élaguer* ne sont pas adaptées à notre contexte multidimensionnel puisque le nombre de combinaisons possibles entre les items est trop important pour pouvoir garantir un passage à l'échelle (surgénération de candidats trop importante). Le paradigme “pattern growth" introduit par [PHMA+04] permet d'extraire les séquences fréquentes de manière gloutonne en parcourant en profondeur l'espace de recherche. Ainsi, l'extraction des motifs se fait en concaténant à la séquence traitée (appelée *séquence préfixe*) les items fréquents sur la base de données projetée par rapport à cette séquence préfixe. Une base de données projetée par rapport à une séquence préfixe est une base de données réduite qui ne contient que les éléments pouvant permettre d'étendre la séquence préfixe.

Afin de faciliter l'écriture des prochaines définitions, nous utilisons le terme de g-k-séquence pour les séquences composées de k items au sein de g itemsets.

Définition 3.5 (g-k-Séquence)**.** Une g-k-séquence S est une séquence composée de g itemsets et de k items de la forme :
$S = \langle\{e_1^1, e_2^1, \ldots, e_{k_1}^1\}, \{e_1^2, e_2^2, \ldots, e_{k_2}^2\}, \ldots, \{e_1^g, e_2^g, \ldots, e_{k_g}^g\}\rangle$ où $\sum_1^g(k_i) = k$.

Par exemple, la séquence $\langle\{(a_1, b_1, *), (a_2, b_2, c_2)\}\{(*, b_2, c_2)\}\rangle$ est une 2-3-séquence.

Lorsqu'on considère des séquences d'itemsets, l'opération de concaténation d'un item e' à une g-k-séquence préfixe $S' = s_1, s_2, \ldots, s_g$ peut s'effectuer de deux façons différentes dans une approche pattern-growth :

- concaténation *inter itemset* où l'item est inséré dans un nouvel itemset (le $(g+1)^{ème}$ itemset de la séquence) : $S' = s_1, s_2, \ldots s_g, \{e'\}$.
- concaténation *intra itemset* où l'item est inséré dans le dernier itemset de la séquence (le $g^{ème}$ itemset de la séquence) : $S' = s_1, s_2, \ldots s_g \cup \{e'\}$.

Ordonner les items au sein des itemsets constitue l'un des moyens d'améliorer le processus d'extraction en éliminant de façon efficace des cas déjà examinés. Notons que la valeur joker * n'existe pas comme valeur réelle dans la base de données. Cette valeur est un méta-symbole qui est inféré à partir des valeurs réellement présentes dans la base de données. Ainsi, les solutions proposées dans un contexte classique par [YHA03] (CloSpan) et [WHL07] (BIDE) ne sont pas directement applicables au contexte multidimentionnel avec valeur joker. Nous illustrons ceci à partir des deux séquences de données présentes dans le tableau Tab. 3.3.

Puisque la valeur joker n'est pas explicitement présente dans les n-uplets, il n'est pas possible de définir un ordre lexicographique total. Ainsi pour les méthodes indiquées précédemment, il n'est pas possible d'obtenir la séquence $\langle\{(a_1, b_2), (*, b_1)\}\rangle$. Ainsi, CloSpan extrait l'item (a_1, b_2) avec un support de 2 et construit ensuite la base projetée à partir de la séquence $\langle\{(a_1, b_2)\}\rangle$ qui contient les séquences $\langle\{\}\rangle$ et $\langle\{(a_2, b_1)\}\rangle$. L'item $(*, b_1)$ n'apparaîtra donc pas comme fréquent dans cette base projetée alors qu'il

l'est dans la base initiale. Ce contre-exemple trivial met en évidence la nécessité d'ordonner les séquences en prenant en compte le caractère joker (*) comme valeur de dimension possible pour les items.

Nous ne souhaitons pas réaliser cette prise en compte par un pré-traitement sur la base de données par extension à l'ensemble des n-uplets contenant la valeur joker. En effet, en considérant une base avec m dimensions d'analyse et n_i items dans un itemset i, ceci produirait $(2^m - 1) \times n_i$ items et une base d'une taille de $(2^m - 1) \sum_{t_i \in DB} n_{t_i}$. Nous souhaitons traiter cette particularité à la volée *pendant* le processus d'extraction de motifs séquentiels multidimensionnels clos. C'est pourquoi nous introduisons un ordre lexical et les fonctions associées afin de gérer les items contenant au moins une valeur joker durant l'extraction des motifs séquentiels multidimensionnels et non en prétraitement.

Nous définissons pour cela la notion d'*itemset étendu*.

Définition 3.6 (Itemset Etendu)**.** Un itemset est étendu s'il est égal à sa fermeture transitive par rapport à la relation de spécialisation ($\prec_S$).

Les itemsets étendus permettent d'extraire les items contenant des valeurs jokers à partir des itemsets des séquences de données.

Afin d'améliorer la découverte des séquences fréquentes, nous introduisons un ordre *lexicographico-spécifique* (LGS). Cet ordre est un ordre alpha-numérique par rapport à la précision des items (nombre de valeurs * dans l'item). Ainsi, la priorité est donnée aux items les plus spécifiques durant le processus d'extraction. Par exemple, l'itemset étendu $i_1 = \{(a_1, b_1), (a_2, b_1), (a_1, *), (a_2, *), (*, b_1)\}$ est ordonné par rapport à l'ordre *LGS*.

Nous pouvons remarquer qu'un itemset étendu ne respecte pas les propriétés ensemblistes des itemsets. En effet, des items comparables apparaissent dans un itemset étendu puisque l'objectif des itemsets étendus est d'énumérer tous les items (contenant des *) possibles à partir des données qui ne contiennent aucune valeur joker. Nous utilisons néanmoins le terme d'itemset, car ce « non respect » de la définition se situe au niveau des séquences de données et non au niveau des séquences extraites. En effet, les itemsets des séquences extraites respecteront la définition d'itemset.

Nous définissons une fonction qui transforme un itemset de la séquence de données en un itemset étendu.

Définition 3.7 (Fonction LGS-Closure). $LGS\text{-}Closure$ est une application d'un itemset i vers l'itemset étendu de i (fermeture transitive par rapport à la relation de spécialisation) ordonné par rapport à l'ordre LGS $<_{lgs}$.

Exemple 3.3 (LGS-Closure). $LGS\text{-}Closure(\{(a_1, b_1), (a_1, b_2)\}) = \{(a_1, b_1), (a_1, b_2), (a_1, *), (*, b_1),$-$(*, b_2)\}$

La fermeture est illustrée par la figure 3.4. Notons que nous ne retournons pas l'item le plus général $(*, *)$ du treillis. En effet, il est inutile de rechercher cet item qui ne respecte pas la définition d'un item multidimensionnel (au moins une dimension différente de *).

$\langle(a_1, *)\rangle_3$
$\langle(a_2, *), (a_2, *)\rangle_3$
$\langle(a_2, *), (*, b_2)\rangle_3$
$\langle(*, b_1), (a_2, *)\rangle_3$
$\langle(*, b_2), (*, b_2)\rangle_3$
$\langle\{(a_1, b_1), (*, b_2)\}\rangle_2$
$\langle\{(a_1, b_2), (*, b_1)\}, (a_2, *), (a_2, *)\rangle_2$
$\langle\{(a_1, b_2), (*, b_1)\}(*, b_2), (a_2, *)\rangle_2$
$\langle(a_2, b_1), (a_2, *)\rangle_2$
$\langle(a_2, b_1), (*, b_1)\rangle_2$
$\langle(a_2, b_1), (*, b_2)\rangle_2$
$\langle(a_2, *), (a_2, b_2)\rangle_2$
$\langle(a_2, *), (a_1, *)\rangle_2$
$\langle(*, b_1), (a_2, b_2)\rangle_2$
$\langle(*, b_1), (a_1, *)\rangle_2$

TAB. 3.2 – Motifs Séquentiels Clos de support 2 et 3

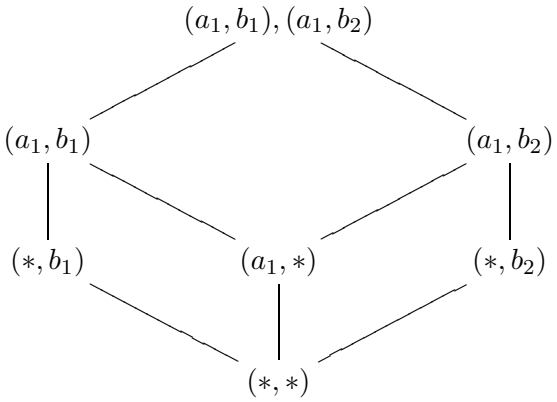


FIG. 3.4 – LGS-Closure

1	$\langle\{(a_1, b_1), (a_1, b_2)\}\rangle$
2	$\langle\{(a_1, b_2), (a_2, b_1)\}\rangle$

TAB. 3.3 – Comment extraire la séquence $\langle\{(a_1, b_2), (*, b_1)\}$?

L'extraction des items fréquents est ainsi effectuée sur chaque itemset étendu des séquences de données. Dans une approche de type pattern-growth, les séquences sont extraites de manière gloutonne, en ajoutant un item fréquent à la séquence préfixe. La séquence préfixe peut être étendue en ajoutant un item fréquent dans un nouvel itemset ou dans le dernier itemset de la séquence préfixe. Il est donc nécessaire de définir un moyen efficace pour étendre une séquence préfixe en ajoutant un item dans le dernier itemset (concaténation intra itemset). De plus, nous devons préserver la notion ensembliste d'un itemset. C'est-à-dire que deux items *comparables* ne peuvent pas apparaître ensemble dans un itemset. Nous définissons ainsi une restriction de la fonction LGS-Closure.

Définition 3.8 (Fonction LGS-Closure$_X$)**.** La fonction $LGS\text{-}Closure_X(i)$ est une application d'un itemset i vers la fermeture de i en filtrant par rapport à l'itemset $X = \{x_1 \leq_{lgs} \ldots \leq_{lgs} x_{k'}\}$ tel que :

$$LGS\text{-}Closure_X(i) = \{e \in LGS\text{-}Closure(i)\ t.q.\ e \geq_{lgs} x_{k'} \text{ et } \nexists x_j \in X \mid x_j \subseteq e\}$$

La fonction LGS-Closure$_X$ énumère tous les items possibles à partir d'un itemset d'une séquence de données tout en filtrant les items déjà extraits (présents dans X) et ceux qui ne permettraient pas de respecter la définition d'itemset.

Exemple 3.4 (LGS-Closure$_X$)**.**

$$LGS\text{-}Closure_{\{(a_1,b_1)\}}(\{(a_1, b_1), (a_1, b_2), (a_1, b_3)\}) = \{(*, b_2), (*, b_3)\}$$

Les items $(a_1, *)$ et $(*, b_1)$ ne sont pas retournés puisque l'ajout de tels items dans l'itemset $\{(a_1, b_1)\}$ ne respecte pas la définition d'itemset multidimensionnel ($(a_1, b_1) \subset (a_1, *)$ et $(a_1, b_1) \subset (*, b_1)$.

Les algorithmes 5 et 6 décrivent l'extraction gloutonne des séquences fréquentes. La méthode *locally frequent items()* permet d'extraire les items fréquents sur la base projetée par la séquence préfixe examinée. Les extensions *intra-itemsets* sont extraites à l'aide de la fonction LGS-Closure$_X$ où X représente le dernier itemset de la séquence préfixe. Les extensions *inter-transactions* sont extraites à l'aide de la fonction $LGS\text{-}Closure$ sans filtre. Ces algorithmes sont les « squelettes » des algorithmes que nous allons définir pour l'extraction de motifs séquentiels multidimensionnels clos.

Au lieu de parcourir plusieurs fois la base de données dans son intégralité, niveau par niveau, comme les approches basées sur le paradigme *a priori*, la base de données est projetée par rapport à la séquence préfixe. Cette projection est assez différente de celle définie dans [PHMA$^+$04]. En effet, puisque nous devons prendre en compte les items contenant des valeurs jokers qui ne sont pas explicitement présentes dans les données, la base de données projetée doit prendre la transaction (itemset du bloc de la séquence

de données) où l'item a été trouvé, et pas seulement l'item lui même comme dans [PHMA+04] où il existe un ordre total entre les items des itemsets des séquences de données. Pour prendre en compte cet itemset, nous utilisons la fonction *LGS-Closure* en filtrant les items déjà extraits.

Par exemple, considérons la séquence de données suivante $S = \langle (a_1, b_1), \{(a_1, b_2), (a_1, b_3)\}, (a_2, b_2)\rangle$ et la séquence préfixe $\alpha = \langle (a_1, b_2)\rangle$. D'après [PHMA+04], la séquence de données projetée par rapport à la séquence préfixe α est $S|\alpha = \{\_(a_1, b_3)\}, (a_2, b_2)\rangle$ où $\_(a_1, b_3)$ indique que l'item (a_1, b_3) et le dernier itemset de α partage le même itemset dans la séquence de données. Afin de prendre en compte les valeurs jokers, notre projection est légèrement différente. La séquence de données projetée est construite de la manière suivante : $S|\alpha = \_LGS\text{-}Closure_{\{(a_1,b_2)\}}(\{(a_1, b_2)(a_1, b_3)\}), \{(a_2, b_2)\}$, ce qui donne la séquence projetée $S|\alpha = \{\_(a_1, b_3)\_(*, b_3)\}, \{(a_2, b_2)\}$ (l'item $\_(a_1, *)$ ne peut pas être dans le même itemset que (a_1, b_2) puisque $(a_1, b_2) \subset (a_1, *)$).

Algorithme 5 : Enumération des séquences fréquentes

Entrées : Base de données DB, seuil de support minimum $minsup$

Sorties : L'ensemble des séquences fréquentes FS

début

```
/* Initialisation                                                    */
```

$Set\ FS \leftarrow \{\}$;

Appel Frequent-Sequences($DB, \langle\rangle, minsup, FS$);

retourner FS;

fin

Algorithme 6 : Frequent-sequences

Entrées : Base de données projetée $DB|_{s_p}$, séquence préfixe s_p, seuil de support minimum $minsup$

Sorties : L'ensemble courant des séquences fréquentes FS

début

si $s_p \neq \langle\rangle$ **alors**
  $FS \leftarrow FS \cup s_p$;

$LF\_s_p = locally\ frequent\ items(DB|_{s_p}, s_p, minsup)$;

si $LF\_s_p = \emptyset$ **alors**
  retourner ;

sinon

  pour chaque *item localement fréquent* $i \in LF\_s_p$ **faire**

    $s'_p = \langle S_p, i\rangle$;

    $DB|_{s'_p}$ =pseudo_projected_database($DB|_{s'_p}, s'_p, minsup, FS$);

    Appel Frequent-Sequences($DB|_{s'_p}, s'_p, minsup, FS$);

fin

3.4.2 $CMSP\_Cand$

Nous présentons dans cette partie un algorithme adapté de CloSpan.

Les motifs séquentiels multidimensionnels clos sont extraits à l'aide des algorithmes 7 ($CMSP\_Cand$) et 8 (SequenceGrowing) en suivant un parcours en profondeur de l'espace de recherche.

L'utilisation de bases de données projetées permet d'éviter de parcourir inutilement des données déjà examinées. En effet, si l'on considère une séquence fréquente α et la séquence préfixe β telles que $\beta \prec_S \alpha$ ou $\alpha \prec_S \beta$ et $DB|_\alpha = DB|_\beta$. Il est inutile de continuer l'exploration de la séquence préfixe β, il suffit de copier le sous-arbre (déjà extrait) de la séquence α à la séquence β comme illustré dans les figures 3.2 et 3.3.

Nous pouvons remarquer que :

- Si $\alpha \prec_S \beta$ alors α ne peut pas être close,
- Si $\beta \prec_S \alpha$ alors les séquences de préfixe β sont déjà connues, ce qui nous permet de ne pas continuer l'exploration des suffixes de β.

Dans le dernier cas, nous pouvons noter que β ne peut pas être close. Cependant, il est nécessaire de garder cette séquence car elle peut être incluse dans d'autres, ce qui permet d'éviter des passes inutiles sur la base de données.

L'algorithme 9 permet d'extraire les items fréquents sur une base de données projetée par rapport à une séquence préfixe. La recherche des items fréquents s'appuie sur la fonction *LGS-Closure* (définition 3.8). La base données projetée est parcourue une seule fois afin d'extraire tous les items fréquents.

Deux types d'items peuvent être extraits :

1. Les items qui ne peuvent pas être inclus dans le dernier itemset de la séquence préfixe ς. Ces items sont ajoutés dans un nouvel itemset de la séquence ς (extension inter itemset). Pour extraire de tels items tout en prenant la valeur joker en compte, nous devons étendre les itemsets des séquences de données de la base de données projetée, pas à pas, à l'aide la fonction *LGS-Closure*.
2. Les items qui peuvent être inclus dans le dernier itemset de la séquence préfixe ς (extension intra itemset). Dans ce cas, nous utilisons la fonction *LGS-Closure* avec en paramètre le dernier itemset de ς.

La dernière étape de l'algorithme 7 vise à éliminer les séquences multidimensionnelles non-closes de l'ensemble des séquences closes candidates. Ce problème consiste à vérifier pour chaque séquence candidate ς s'il existe une séquence candidate ς' telle que $\varsigma \prec_S \varsigma'$ et $support(\varsigma) = support(\varsigma')$. Un algorithme naïf qui compare chaque séquence avec les autres est relativement coûteux puisqu'il est quadratique par rapport au nombre de séquences closes candidates. Ce qui rend impossible son utilisation dès que l'ensemble des candidats est important. Nous adoptons l'algorithme de vérification rapide par subsomption introduit par [ZH02]. Si la valeur de support est dense, alors le *support* ne peut pas être une clé de hachage pertinente, qui garantie une distribution homogène des clés. [ZH02] propose d'utiliser la somme des identifiants des séquences (noté $\tau(D_S)$) comme clé de hachage au lieu du support. Toutefois,

dans un contexte d'extraction de séquence, l'équivalence de $\tau(D_S)$ n'implique pas nécessairement une équivalence de support. Ainsi, pour les séquences multidimensionnelles qui partagent le même $\tau(D_S)$, nous devons vérifier leur support afin d'éliminer les candidats non-pertinents. Cette clé de hachage, également utilisé dans CloSpan, est facile à calculer. De plus, elle permet de réduire le nombre de candidats à vérifier pour une séquence. Ainsi la complexité de la recherche des séquences closes parmi un ensemble de candidats est $\Theta(\sum n_{\tau_i}^2)$ où n_{τ_i} est le nombre de séquences candidates qui partagent le même τ_i. n_{τ_i} est significativement plus petit que le nombre total de séquences candidates ($\sum n_{\tau_i}$).

Algorithme 7 : $CMSP\_Cand$

Entrées : Base de données DB, seuil de support minimum $minsup$

Sorties : L'ensemble des motifs séquentiels multidimensionnels clos C

début

```
/* Initialisation                                                    */
Set L ← {}
Set C ← {}
Sequence α ← ⟨⟩
/* Recherche des séquences fréquentes en profondeur d'abord          */
SequenceGrowing(α, DB, L, minsup)
/* Suppression des séquences non-closes L                            */
pour chaque s1 ∈ L faire
    pour chaque s2 ∈ L faire
        si s1 ≺S s2 et support(s1) = support(s2) alors
            delete(s1, L)
C ← L
retourner C
```

fin

Exemple 3.5. Déroulons l'algorithme $CMSP\_Cand$ sur la base de données DB illustrée Tab. 3.4 avec un seuil de support minimum fixé à 2.

L'algorithme principal **CMSP_Cand** appelle l'algorithme **SequenceGrowing** avec comme paramètres la séquence vide $\langle\rangle$, DB et $minsup = 2$.

La première étape consiste à extraire tous les items fréquents sur DB à l'aide **getFrequentItems** :

$\{(a_1, b_1, c_1)_2, (a_1, b_2, c_3)_2, (a_1, b_1, *)_2, (a_1, b_2, *)_2, (a_1, b_3, *)_2, (a_1, *, c_1)_2, (a_1, *, c_2)_2, (a_1, *, c_3)_2, (*, b_1, c_1)_2, (*, b_2, c_3)_2, (a_1, *, *)_2, (*, b_1, *)_2, (*, b_2, *)_2, (*, b_3, *)_2, (*, *, c_1)_2, (*, *, c_2)_2, (*, *, c_3)_2\}$

Les séquences sont extraites en parcourant l'espace de recherche en profondeur d'abord en fonction de l'ordre *LGS*.

Les séquences de préfixe $\langle(a_1, b_1, c_1)\rangle$ sont alors recherchées sur la base projetée $DB|\langle(a_1, b_1, c_1)\rangle$.

Algorithme 8 : SequenceGrowing

Entrées : Séquence préfixe α, base de données projetée $DB|_\alpha$, ensemble des clos candidats L , seuil de support minimum $minsup$

Sorties : Ensemble des séquences de préfixe α

début

```
/* α est potentiellement close                                    */
insert(α, L);
/* Test si élagage possible                                       */
si ∃β | (α ≺_S β or β ≺_S α) et ils partagent la même base projetée alors
    Copier les descendants de β dans α;
    retourner
Set F_l ← getFrequentItems(DB|α, minsup);
pour chaque α' ← α.b faire
    Construire DB|_α';
    SequenceGrowing(α', DB|_α', L, minsup);
```

fin

Algorithme 9 : getFrequentItems

Entrées : Base de données projetée $DB|_\alpha$, Seuil de support minimum $minsup$

Sorties : Ensemble F_l des items localement fréquent sur $DB|_\alpha$

début

```
/* Pour chaque séquence de données S_i de DB|_, nous avons :
   S_i = LGS-Closure_{lastItemset(α)}(same).otherTrans            */
/* Nous devons examiner toutes les séquence de données de DB|α    */
pour chaque S_i ∈ DB|_α faire
    pour chaque item _e ∈ same faire
        gérer _e ;
    pour chaque itemset is in other faire
        /* Recherche des extensions inter itemset                 */
        SearchOtherTransFrequentItem e dans LGS-Closure(is);
        /* Recherche des extensions intra-itemset                 */
        si is supporte lastItemset(α) alors
            SearchSameTransFrequentItem _e dans LGS-Closure_{lastItemset(α)}(is);
retourner (F_l = {e|support(e) ≥ suppmin});
```

fin

B_1	$\langle\{(a_1, b_2, c_3)\}\{(a_1, b_1, c_1), (a_1, b_3, c_2)\}\rangle$
B_2	$\langle\{(a_1, b_2, c_2)(a_1, b_2, c_3)\}\{(a_1, b_1, c_1), (a_1, b_3, c_4)\}\{(a_1, b_1, c_1)\}\rangle$

TAB. 3.4 – DB

Les séquences $\langle\{(a_1, b_1, c_1), (a_1, b_3, *)\}\rangle$ et $\langle\{(a_1, b_1, c_1), (*, b_3, *)\}\rangle$ sont découvertes. Aucun item fréquent n'apparaît sur les bases projetées par rapport aux séquences préfixes $\langle\{(a_1, b_1, c_1), (a_1, b_3, *)\}\rangle$ et $\langle\{(a_1, b_1, c_1), (*, b_3, *)\}\rangle$.

La recherche des séquences fréquentes s'effectue donc à partir de la séquence préfixe $\langle(a_1, b_2, c_3)\rangle$. Lorsque la séquence préfixe $\langle\{(a_1, b_2, c_3)\}\{(a_1, b_1, c_1)\}\rangle$ est considérée. On détecte que les séquences $\langle\{(a_1, b_2, c_3)\}\{(a_1, b_1, c_1)\}\rangle$ et $\langle(a_1, b_1, c_1)\rangle$ partagent la même base de données projetée. Ainsi, il est inutile de continuer l'exploration de la séquence préfixe $\langle\{(a_1, b_2, c_3)\}\{(a_1, b_1, c_1)\}\rangle$. En effet, les séquences $\langle\{(a_1, b_2, c_3)\}, \{(a_1, b_1, c_1), (a_1, b_3, *)\}\rangle$, et $\langle\{(a_1, b_2, c_3)\}\{(a_1, b_1, c_1), (*, b_3, *)\}\rangle$ sont découvertes sans parcours supplémentaire de l'espace de recherche à partir des séquences déjà extraites de préfixe $\langle(a_1, b_1, c_1)\rangle$.

La recherche des séquences fréquentes continue avec la séquence préfixe $\langle(a_1, b_1, *)\rangle$.

Le processus est réitéré jusqu'à l'extraction des séquences de préfixe $\langle(*, *, c_3)\rangle$.

Finalement, les séquences closes sont recherchées dans l'ensemble des clos candidats. Seules les séquences $\langle\{(a_1, b_2, c_3)\}\{(a_1, b_1, c_1), (a_1, b_3, *)\}\rangle$ et $\langle\{(a_1, *, b_2)\}\rangle$ sont retournées.

3.4.3 $CMSP\_Free$

L'algorithme précédent d'extraction de motifs séquentiels multidimensionnels doit gérer un ensemble de candidats (ensemble L). Il doit ensuite calculer, en post-traitement, les séquences closes parmi les séquences candidates. Cette gestion est relativement coûteuse (quadratique en la taille de l'ensemble candidat) même s'il existe des techniques qui permettent de réduire son coût. Dans cette partie, nous proposons un algorithme qui ne s'appuie pas sur la gestion d'un ensemble de candidats. Cette approche est basée sur Bide [WHL07]. Nous détaillons les définitions préliminaires avant de présenter l'algorithme associé.

Extensions et clos

Actuellement, la plupart des algorithmes d'extraction de motifs clos ont besoin de maintenir l'ensemble des clos (ou juste candidats) en mémoire et vérifier en post-traitement si un motif peut être absorbé ou non par un autre motif. Mais la maintenance d'un tel ensemble est très coûteuse (quadratique en la taille de l'ensemble des clos candidats), c'est pourquoi notre objectif est d'éviter une telle gestion.

Tout d'abord, revenons à la définition même d'un motif séquentiel multidimensionnel clos. Si une g-k-séquence $S = s_1, \ldots, s_g$ n'est pas close alors il existe une séquence S' de même support telle que

$S \prec_S S'$. La définition 3.9 présente les cinq différents types de construction d'une séquence plus spécifique à partir d'une séquence préfixe.

Définition 3.9. Une séquence plus spécifique S' peut être construite de cinq façons différentes à partir d'une g-k-séquence préfixe $S = \langle s_1, s_2, \ldots, s_g \rangle$:

- extension vers l'avant inter itemset $S' = \langle s_1, s_2, \ldots, s_g, \{e'\} \rangle$,
- extension vers l'avant intra itemset $S' = \langle s_1, s_2, \ldots s_g \cup \{e'\} \rangle$,
- extension vers l'arrière inter itemset $S' = \langle s_1, s_2, \ldots, s_i, \{e'\}, s_{i+1}, \ldots, s_g \rangle$,
- extension vers l'arrière intra itemset $S' = \langle s_1, s_2, \ldots s_i \cup \{e'\}, s_{i+1}, \ldots, s_g \rangle$,
- spécialisation d'un item si $\exists i \in \{1, \ldots, g\}, \exists e, \exists e' \ tq \ e \prec_S e'$:
  $S' = \langle s_1, s_2, \ldots, s_{i-1}, s_i[e'/e], s_{i+1}, \ldots, s_g \rangle$ où $s_i[e'/e]$ correspond à la substitution de e par e' dans s_i.

Nous verrons que le dernier point peut être facilement détecté grâce à l'ordre du parcours dès lors que les précédents le sont.

Théorème 1 (Extension bi-directionnelle)**.** Une séquence S est close si et seulement si elle n'accepte aucune extension vers l'avant, extension vers l'arrière, et spécialisation, qui préserve son support.

Démonstration. La démonstration découle trivialement de la définition même des motifs clos (Déf. 3.3). □

A partir du théorème 1, nous savons que pour déterminer si une séquence préfixe est close, nous devons vérifier si elle ne peut pas avoir d'extension vers l'avant ou vers l'arrière ainsi qu'une spécialisation d'item qui préserve le support de la séquence préfixe. Il est relativement facile de trouver des extensions vers l'avant à partir du lemme suivant.

Lemme 3.1 (Extension vers l'avant)**.** Pour une séquence S, l'ensemble complet des extensions vers l'avant est équivalent à l'ensemble des items localement fréquents sur la base projetée par rapport à S ayant un support égal à $support(S)$.

Démonstration. Les items localement fréquents sont extraits en parcourant la base de données projetée par rapport à la séquence préfixe S_p. Puisque chaque événement apparaît pendant ou après la séquence préfixe S_p, s'il existe dans toutes les séquences de données projetées de la base de données, alors il forme une extension vers l'avant. Tout événement apparaissant après la première instance de S_p est inclus dans la base de données projetée, ce qui signifie que l'ensemble complet des extensions vers l'avant peut être extrait en parcourant la base de données projetée par rapport à S_p. □

Pour les extensions vers l'arrière, la recherche d'extensions est beaucoup moins triviale. En effet, une extension vers l'arrière peut être réalisée de deux façons différentes :

- $S' = s_1, s_2, \ldots, s_i, \{e'\}, s_{i+1}, \ldots, s_g$

– $S' = s_1, s_2, \ldots s_i \cup \{e'\}, s_{i+1}, \ldots, s_g$

Soit un item s'insère dans un nouvel itemset, entre deux itemsets s_i et s_{i+1} existants, soit il s'insère dans un itemset existant. On peut caractériser ces insertions vers l'arrière par des insertions vers l'arrière *inter*-itemsets ou *intra*-itemsets.

Comme une séquence peut se répéter plusieurs fois à l'intérieur d'une séquence de données, on peut identifier g intervalles pour localiser les possibles insertions vers l'arrière d'une g-k-séquence. La figure 3.5 représente ces g intervalles.

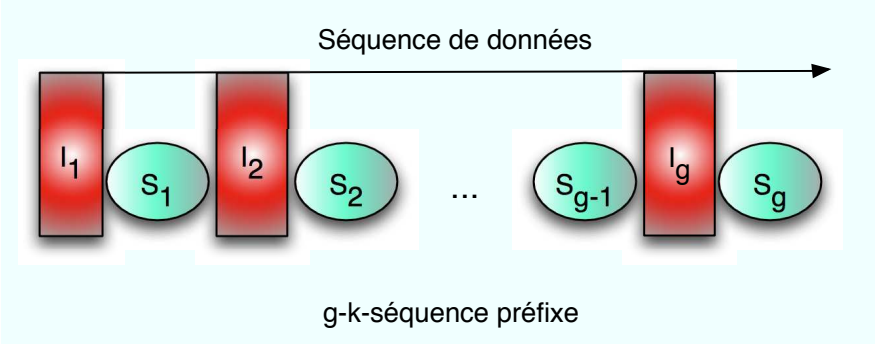


FIG. 3.5 – Les différents intervalles d'insertion possibles pour les extensions vers l'arrière d'une g-k-séquence préfixe $S_p = \langle s_1, s_2, \ldots, s_g \rangle$ au sein d'une séquence de données

Il faut *maximiser* ces intervalles afin de détecter toutes les extensions possibles vers l'arrière.

Définition 3.10 ($i^{ème}$ Intervalle maximal)**.** Etant données une g-k-séquence préfixe $S_p = \langle s_1, s_2, \ldots, s_g \rangle$ et une séquence de données S, le $i^{ème}$ intervalle maximal se définit de la façon suivante :

- pour $i = 1$: la sous-séquence du début de S jusqu'à strictement avant $da(s_1)$ la dernière apparition de s_1 dans S telle que $da(s_1) < da(s_2) < \ldots < da(g)$
- pour $1 < i \leq g$: la sous-séquence entre la première apparition de la séquence $\langle s_1, s_2, \ldots, s_{i-1} \rangle$ notée $pa(\langle s_1, s_2, \ldots, s_{i-1} \rangle)$ et strictement avant $da(s_i)$ telle que $da(s_i) < da(s_{i+1}) < \ldots < da(g)$

Exemple 3.6. Le tableau Tab. 3.5 illustre les intervalles maximaux de plusieurs séquences sur la base de données DB décrite dans le tableau Tab. 3.4.

Les $i^{èmes}$ intervalles maximaux sont la base de la détection des extensions vers l'arrière définies dans le lemme suivant.

Lemme 3.2 (Vérification d'une extension vers l'arrière)**.** Soit la g-k-séquence préfixe $S_p = \langle s_1, s_2, \ldots, s_g \rangle$, s'il existe un entier i tel qu'il existe un item e qui apparaît dans chacun des $i^{èmes}$ intervalles maximaux de la séquence de préfixe S_p dans DB, alors e est une extension vers l'arrière.

Autrement, si nous ne pouvons pas exhiber d'item e qui apparaît dans chacun des $i^{ème}$ intervalles maximaux, alors il ne peut pas y avoir d'extension vers l'arrière de la séquence préfixe S_p dans la base de données DB.

DB		B_1 $\langle\{(a_1,b_2,c_3)\}\{(a_1,b_1,c_1),(a_1,b_3,c_2)\}\rangle$ B_2 $\langle\{(a_1,b_2,c_2)(a_1,b_2,c_3)\}\{(a_1,b_1,c_1),(a_1,b_3,c_4)\}\{(a_1,b_1,c_1)\}\rangle$
Séquence	i	$i^{ème}$ Intervalle maximal de DB
$\langle(a_1,b_1,c_1)\rangle$	1	$\langle\{(a_1,b_2,c_3)\}\rangle$ $\langle\{(a_1,b_2,c_2)(a_1,b_2,c_3)\}\{(a_1,b_1,c_1),(a_1,b_3,c_4)\}\rangle$
$\langle(a_1,b_2,c_3)\rangle$	1	$\langle\rangle$ $\langle\rangle$
$\langle\{(a_1,b_2,c_3)\}\{(a_1,b_1,c_1)\}\rangle$	2	$\langle\rangle$ $\langle\{(a_1,b_1,c_1),(a_1,b_3,c_4)\}\rangle$

TAB. 3.5 – Exemple d'intervalles maximaux sur DB (Tab. 3.4)

Démonstration. Soit la g-k-séquence préfixe $S_p = \langle s_1, s_2, \ldots, s_g\rangle$. S'il existe un item e qui apparaît dans chacun des $i^{èmes}$ intervalles maximaux de la séquence de préfixe S_p dans DB, alors on peut construire la séquence $S' = \langle s_1, s_2, \ldots, s_{i-1} \cup \{e\}, s_i, \ldots, s_g\rangle$ ou $S' = \langle s_1, s_2, \ldots, s_{i-1}, \{e\}, s_i, \ldots, s_g\rangle$ telle que $S_p \prec_S S'$ et $supp(S') = supp(S_p)$. Donc e est une extension vers l'arrière.

Supposons qu'il existe une séquence $S'_p = \langle s_1, s_2, \ldots, s_{i-1} \cup \{e\}, s_i, \ldots, s_g\rangle$ ou $S'_p = \langle s_1, s_2, \ldots, s_{i-1}$-$, \{e\}, s_i, \ldots, s_g\rangle$ qui absorbe S_p. Dans chaque séquence de données de DB contenant S_p, l'item e' doit apparaître après la première apparition de $\langle s_1, \ldots, s_{i-1}\rangle$ et avant la dernière apparition de la sous-séquence $\langle s_i, \ldots, s_g\rangle$, ce qui signifie que e doit apparaître dans chacun des $i^{èmes}$ intervalles maximaux de S_p dans DB. Ainsi, si nous ne pouvons pas exhiber d'items apparaissant dans chaque $i^{ème}$ intervalle de S_p alors il ne peut pas y avoir d'extension vers l'arrière.

□

Comme indiqué dans le tableau Tab. 3.5, l'item (a_1, b_2, c_3) apparaît dans chacun des premiers intervalles maximaux de la séquence $\langle(a_1, b_1, c_1)\rangle$ dans la base de données DB. L'item (a_1, b_2, c_3) est donc une extension vers l'arrière de la séquence $\langle(a_1, b_1, c_1)\rangle$ qui ne peut donc pas être close puisqu'on peut construire une séquence plus spécifique $\langle\{(a_1, b_1, c_1)\}\{(a_1, b_2, c_3\ )\}\rangle$ de même support.

Une séquence préfixe ne peut également pas être close s'il existe une spécialisation d'un item de la séquence préfixe. L'ordre LGS, que nous adoptons, nous permet d'extraire les séquences closes en commençant par celles qui contiennent les items les plus spécifiques (le moins de valeurs $*$). Ainsi, s'il existe une spécialisation possible d'une séquence préfixe considérée, alors la "séquence spécialisée", qui contient au moins un item plus spécifique, sera déjà présente dans l'ensemble des clos déjà extraits. Ainsi, si une séquence est potentiellement close (pas d'extensions vers l'avant ou l'arrière), il suffit de vérifier qu'il n'existe pas de séquence plus spécifique dans l'ensemble des séquences closes déjà extraites. Notons que cet ensemble est sensiblement plus petit que l'ensemble des séquences fréquentes. Cette opération

de vérification n'est donc pas trop coûteuse. Dans le pire des cas, on doit considérer toutes les séquences closes déjà extraites dont le support est égal à la séquence préfixe examinée.

Elagage de l'espace de recherche

Tout en recherchant les nouvelles séquences fréquentes avec l'algorithme d'énumération des séquences, nous pouvons utiliser la propriété de fermeture bidirectionnelle pour vérifier si la séquence préfixe est close dans le but de générer un ensemble non redondant de connaissances.

Bien que la propriété de fermeture retourne un ensemble plus compact, cela ne permet pas d'extraire les séquences plus efficacement. Par exemple, il peut n'y avoir aucun clos au delà d'un certain nœud dans l'arbre des préfixes, il faudrait donc éviter de parcourir inutilement la branche et réduire ainsi significativement l'espace de recherche.

Comme nous l'avons dit précédemment, une séquence peut apparaître plusieurs fois dans une séquence de données. Dans la définition 3.10, nous avons introduit la notion d'intervalle maximal afin de pouvoir détecter toutes les extensions vers l'arrière. Nous désirons maintenant *minimiser* ces intervalles afin de détecter les séquences préfixes « non-prometteuses ». Nous définissons ainsi la notion d'$i^{ème}$ intervalle minimal.

Définition 3.11 ($i^{ème}$ intervalle minimal)**.** Pour une séquence de données S contenant une g-k-séquence préfixe $S_p = \langle s_1, s_2, \ldots, s_g \rangle$, l'$i^{ème}$ intervalle minimal de S_p dans S se définit de la façon suivante :

- Si $i = 1$ alors c'est la sous-séquence située strictement avant la première apparition de s_1.
- Si $1 < i \leq g$ alors c'est la sous-séquence comprise entre la première apparition de la séquence $\langle s_1, \ldots, s_{i-1} \rangle$ et strictement avant $pa(s_i)$ telle que $pa(s_i) < pa(s_{i+1}) \leq \ldots \leq pa(s_g)$.

Exemple 3.7. Le tableau Tab. 3.6 illustre les intervalles minimaux de plusieurs séquences sur la base de données DB décrite Tab. 3.4.

DB		B_1 $\langle\{(a_1,b_2,c_3)\}\{(a_1,b_1,c_1),(a_1,b_3,c_2)\}\rangle$ B_2 $\langle\{(a_1,b_2,c_2)(a_1,b_2,c_3)\}\{(a_1,b_1,c_1),(a_1,b_3,c_4)\}\{(a_1,b_1,c_1)\}\rangle$
Séquence	i	$i^{ème}$ Intervalle minimal de DB
$\langle(a_1,b_1,c_1)\rangle$	1	$\langle\{(a_1,b_2,c_3)\}\rangle$ $\langle\{(a_1,b_2,c_2)(a_1,b_2,c_3)\}\rangle$
$\langle(a_1,b_2,c_3)\rangle$	1	$\langle\rangle$ $\langle\rangle$
$\langle\{(a_1,b_2,c_3)\}\{(a_1,b_1,c_1)\}\rangle$	2	$\langle\rangle$ $\langle\rangle$

TAB. 3.6 – Exemple d'intervalles minimaux sur DB (Tab. 3.4)

Théorème 2 (Elagage). Soit la g-k-séquence préfixe $S_p = \langle s_1, s_2, \ldots, s_g \rangle$, s'il existe un entier i tel qu'il existe un item e qui apparaît dans chacun des $i^{èmes}$ intervalles minimaux de S_p dans la base de données DB, alors il ne peut plus y avoir de séquence close de préfixe S_p.

Démonstration. Si un item e apparaît dans chacun des intervalles minimaux de la g-k-séquence préfixe S_p alors nous pouvons utiliser la nouvelle séquence préfixe S'_p contenant l'item e. En effet, $S_p \prec_S S'_p$ et $support(S'_p) = support(S_p)$. Ainsi tout item localement fréquent sur la base projetée par S_p est aussi fréquent sur la base projetée par S'_p. Ce qui implique qu'il n'y a aucun espoir d'extraire un motif clos de préfixe S_p, le parcours de la séquence préfixe S_p peut donc être interrompu. □

A partir de ce théorème, nous pouvons par exemple arrêter d'explorer les séquences de préfixe $\langle (a_1, b_1, c_1) \rangle$. En effet, l'item (a_1, b_2, c_3) apparaît dans chacun des premiers intervalles minimaux de $\langle (a_1, b_1, c_1) \rangle$ dans DB, comme indiqué Tab. 3.6. Ainsi toutes les séquences de préfixe $\langle (a_1, b_1, c_1) \rangle$ n'ont aucune raison d'être explorées puisqu'elles seront découvertes à l'aide de la séquence préfixe $\langle \{(a_1, b_2, c_3)\}\{(a_1, b_1, c_1)\} \rangle$.

Grâce aux théorèmes et définitions introduits précédemment, nous pouvons maintenant écrire les algorithmes permettant la mise en œuvre de l'extraction des motifs séquentiels multidimensionnels clos sans gestion d'ensemble de candidats.

Algorithmes

Les algorithmes 10 et 11 décrivent l'extraction des motifs séquentiels clos sans gestion d'ensembles de candidats. Ces algorithmes conservent la structure des algorithmes d'extraction de séquences fréquentes. En effet, dans le pire des cas, l'espace de recherche est le même. Toutefois, nous introduisons une condition d'élagage qui permet de réduire l'espace de recherche. L'algorithme 11 présente le cœur de l'extraction des motifs séquentiels multidimensionnels. Dans une première étape, si le nombre des extensions vers l'avant et vers l'arrière de la séquence préfixe S_p est nul, alors il faut vérifier qu'il n'existe pas de séquence plus spécifique dans l'ensemble FCS des motifs séquentiels multidimensionnels clos déjà extraits. S'il n'en existe aucune de même support que S_p, alors S_p est ajoutée à FCS. L'ensemble FCS est partitionné en sous-ensembles de motifs séquentiels multidimensionnels en fonction de leur support. Ainsi la recherche dans S_p d'une séquence plus spécifique que S_p s'effectue sur un sous-ensemble des motifs séquentiels clos déjà extraits. Dans le pire des cas, la complexité de cette opération de vérification est $O(l_\sigma)$ où l_σ est le nombre de séquences closes déjà extraites de support σ. Ensuite, chaque item localement fréquent e sur la base projetée est considéré. L'algorithme vérifie s'il est possible d'élaguer l'espace de recherche pointé par la séquence préfixe $S_p.e$ (e ajouté dans un nouvel itemset ou non). Si ce n'est pas possible alors l'algorithme calcule le nombre d'extensions vers l'arrière de la séquence préfixe $S_p.e$ et continue de fouiller l'espace de recherche indiqué par la nouvelle séquence préfixe.

3.5 Expérimentations

Dans cette section, nous présentons des résultats d'expérimentations menées sur des jeux de données synthétiques et sur des jeux de données réels. Afin de mettre en relief notre proposition, nous comparons les temps d'exécution de $CMSP\_Cand$ et $CMSP\_Free$.

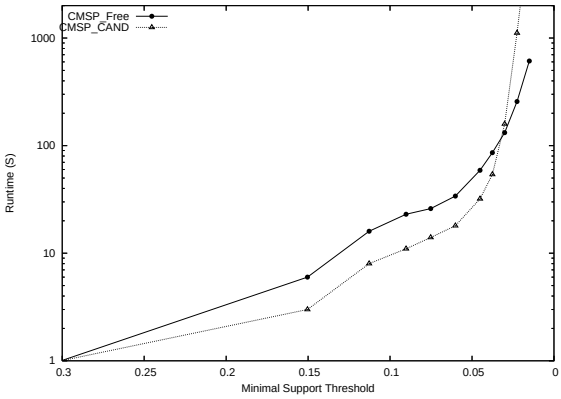
Données Synthétiques Une base de données a été générée par *IBM Quest Market-Basket Synthetic Data Generator*[1] (1000000 enregistrements), où les items (1 dimension) ont été transformés en des items multidimensionnels (3 dimensions). Puisque notre approche effectue à la fois des vérifications vers l'avant et l'arrière pour déterminer si une séquence fréquente est close, nous pouvons supposer que lorsque les données examinées sont éparses (nombre de fréquents faible et similaire au nombre de clos) l'approche basée sur la gestion d'un ensemble de candidats peut être plus rapide. C'est ce qui se passe jusqu'à une certaine valeur du support minimal. Mais, dès que le support considéré entraîne un nombre important de séquences fréquentes, la méthode $CMSP\_Cand$ n'est plus adaptée. En effet, le temps d'exécution d'une telle approche est très sensible au nombre de motifs fréquents puisque la plupart d'entre eux sont considérés comme potentiellement candidats ; leur coût de traitement étant quadratique en la taille de l'ensemble des clos candidats. $CMSP\_Free$ est plus robuste face à ce phénomène puisqu'elle ne considère aucun ensemble de candidats et utilise des propriétés d'élagages supplémentaires qui évite de parcourir inutilement certaines parties de l'espace de recherche.

La courbe de la figure 3.6(c) montre le comportement de $CMSP\_Free$ en fonction de la taille de la base (en nombre de séquences de données). Le temps d'extraction des motifs multidimensionnels clos est proportionnel à la taille de la base. Ce qui nous permet de considérer le passage à l'échelle de notre approche par rapport à ce paramètre.

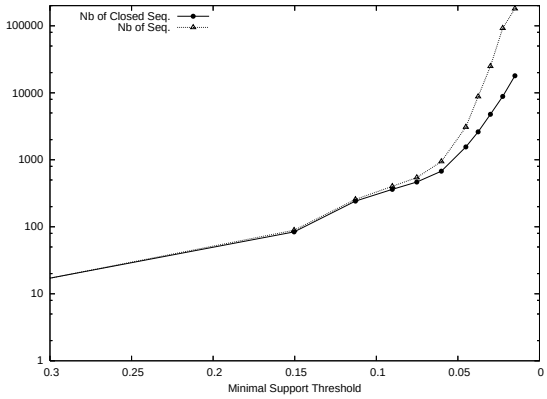
La figure 3.6(d) décrit le temps d'exécution de l'extraction des motifs séquentiels multidimensionnels avec $CMSP\_Free$ en fonction du nombre de dimensions d'analyse. Plus le nombre de dimensions d'analyse augmente, plus la pente de la courbe est forte. Ceci vient du fait que ajouter des dimensions équivaut à agrandir la base de données « en largeur ». Le nombre d'items à considérer devient plus important. Ce paramètre limite l'application de notre proposition avec un grand nombre de dimensions d'analyse. Toutefois, quand le nombre de dimensions d'analyse est trop important, nous pouvons imaginer approximer l'ensemble des motifs séquentiels multidimensionnels.

Cube de données réel Nous avons mené des expérimentations sur un cube de données issu de la base des clients résidentiels d'EDF décrite dans l'annexe A. Nous considérons cinq dimensions d'analyse. Ces expérimentations confortent les résultats obtenus sur les jeux de données synthétiques : dès que le nombre de séquences extraites devient trop important, une approche avec gestion d'un ensemble de

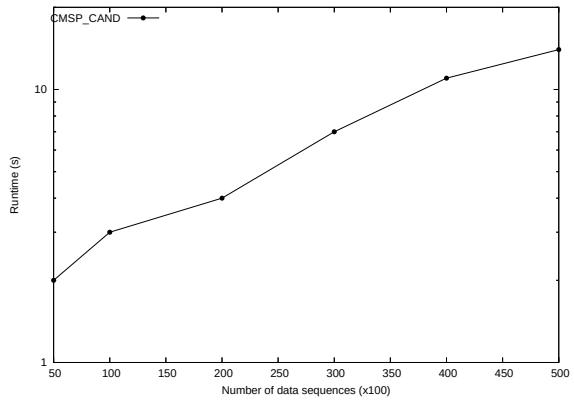
[1] www.almaden.ibm.com/software/quest/Resources/datasets/syndata.html#assocSynData



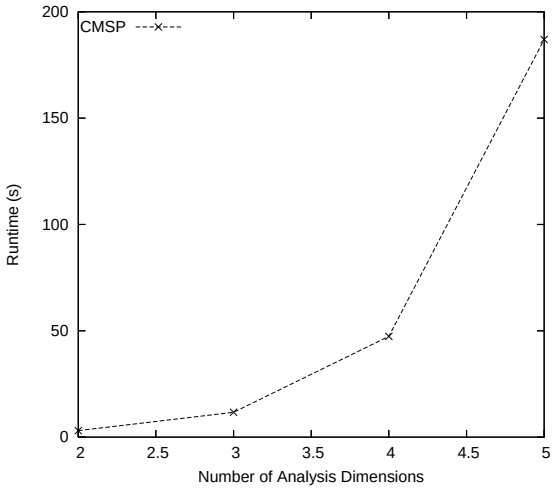
(a) Temps d'exécution en fonction du support



(b) # fréquents et clos en fonction du support



(c) Temps d'exécution en fonction de $|DB|$ ($minsup = 0.2$)



(d) Temps d'exécution en fonction du nombre de dimensions d'analyse ($minsup = 0.2$)

FIG. 3.6 – Expérimentations sur des données synthétiques

clos candidats n'est plus adaptée alors que notre approche ne décroche pas et permet d'extraire des connaissances avec des supports très faibles.

3.6 Discussion

Dans ce chapitre, nous définissons les motifs séquentiels multidimensionnels clos. Nous proposons à la fois un cadre formel pour l'extraction de tels motifs et les algorithmes associés. Extraire des motifs séquentiels multidimensionnels clos permet de disposer d'une représentation condensée des motifs sans perte d'information (motifs, support). Cette représentation permet le calcul de différentes mesures d'intérêt (*e.g.* confiance pour les règles séquentielles) sans passe supplémentaire sur les données puisque tous les supports des séquences fréquentes peuvent être retrouvés à partir des motifs séquentiels multidimensionnels clos. De surcroît, rechercher des motifs clos permet d'introduire des propriétés d'élagages supplémentaires de l'espace de recherche ; ce qui, dans notre contexte, est primordial pour envisager un passage à l'échelle de ces approches. Les travaux de la littérature sur les itemsets clos et les motifs séquentiels clos ne sont pas adaptés à notre contexte du fait de la valeur joker *. Nous proposons une solution originale pour prendre en compte les items contenant une ou plusieurs valeurs jokers car ces items ne sont pas explicitement présents dans les données et doivent être inférés à partir des n-uplets de la base. Nous adoptons le paradigme *pattern growth* pour extraire des motifs séquentiels multidimensionnels clos. Deux algorithmes $CMSP\_Cand$ et $CMSP\_Free$ sont proposés. Des expérimentations sur des jeux de données synthétiques et des jeux de données réels soulignent l'intérêt d'utiliser des représentations condensées dans un contexte multidimensionnel.

Les perspectives associées à cette proposition sont nombreuses. L'utilisation d'autres représentations condensées pourrait permettre d'améliorer l'extraction des séquences fréquentes. De telles représentations (non-dérivable [CG02], k-libre [BBR03]) sont très présentes dans le contexte des itemsets mais il existe encore trop peu de travaux pour les motifs séquentiels ou les motifs multidimensionnels. L'extraction des motifs séquentiels multidimensionnels sous contraintes (top k) peut aussi nous permettre d'élaguer plus rapidement l'espace de recherche.

Algorithme 10 : $CMSP\_FREE$

Entrées : Base de données de séquences DB, seuil de support minimal $minsup$

Sorties : L'ensemble des motifs séquentiels multidimensionnels clos FCS

début

$FCS = \emptyset$;

$F1 =$items-fréquents$f(DB, minsup)$;

pour chaque *1-séquence* $f_1 \in F_1$ **faire**

si *élagage de* f_1 *n'est pas possible* **alors**

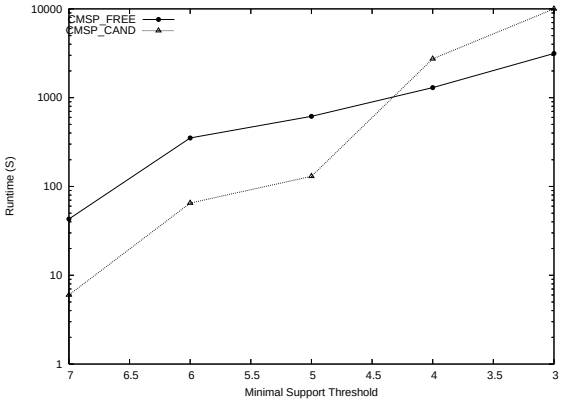
`/* Dénombrement des extensions vers l'arrière. */`

$BEI =$ #*Extension_vers_arrière*$(f1, SDB^{f1})$;

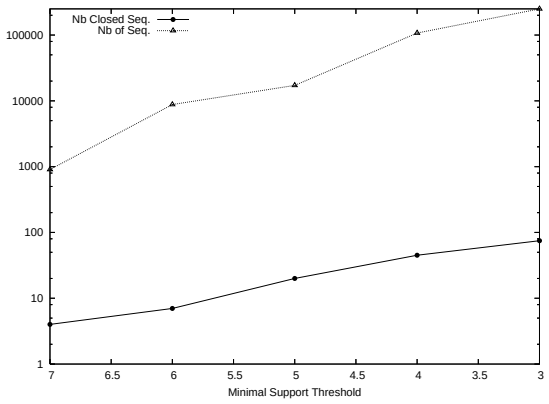
Appel routine $CMSP(DB|_{f1}, f1, minsup, BEI, FCS)$;

retourner FCS;

fin



(a) Temps d'exécution en fonction du support



(b) Temps d'exécution en fonction du support

FIG. 3.7 – Expérimentations sur cube de données réel

Algorithme 11 : routine $CMSP\_FREE$

Entrées : Une base projetée $DB|_{S_p}$, une séquence préfixe S_p, seuil de support minimal $minsup$, le nombre de d'extension vers l'arrière BEI

Sorties : L'ensemble courant des séquences fréquentes closes FCS

début

```
/* Rechercher items fréquents et extensions vers l'avant                    */
```

$LFI =$items-fréquents$(DB|_{S_p}, minsup)$;

$FEI = |\{z \in LFI \ / \ support(z) = support(S_p)\}|$;

si $(BEI + FEI) = 0$ **alors**

```
/* On vérifie s'il n'y a pas de spécialisation déjà présente dans FCS
   */
```

si $(\nexists \alpha \in FCS \mid S_p \prec_S \alpha \wedge supp(\alpha) = supp(S_p))$ **alors**

$FCS = FCS \cup \{S_p\}$;

pour chaque $i \in LFI$ **faire**

```
/* Ajout de l'item fréquent à la séquence (intra ou inter itemset) et
   construction base projetée                                               */
```

$S'_p = \langle S_p.i \rangle$;

$DB|_{S_{p'}} = pseudo\ projected\ database(DB_{S_p}, S'_p)$;

pour chaque $i \in LFI$ **faire**

```
/* On vérifie si un élagage est possible                                    */
```

si *élagage de* S'_p *possible* **alors**

$BEI = backward\ extension\ check(S'_p, DB|_{S'_p})$;

Appel routine $CMSP(DB|_{S'_p}, S'_p, minsup, BEI, FCS)$;

fin

Bilan et Perspectives

Dans cette partie, nous proposons une définition des motifs séquentiels multidimensionnels. L'extraction de motifs séquentiels multidimensionnels permet de mettre en exergue des relations entre des valeurs de différentes dimensions suivant une relation d'ordre (e.g. temps). Ce problème est une généralisation du problème d'extraction de motifs séquentiels « classiques » [AS95] et du problème d'extraction de motifs séquentiels multidimensionnels défini par [PHP+01]. L'espace de recherche associé à l'extraction des motifs séquentiels multidimensionnels est donc plus important et peut compromettre le passage à l'échelle des techniques d'extraction de motifs séquentiels multidimensionnels. Nous proposons deux façons différentes de s'attaquer à ce problème en :

Ciblant des parties de l'espace de recherche : Nous proposons d'extraire des motifs séquentiels multidimensionnels à partir des items les plus spécifiques. Ainsi seules les séquences fréquentes composées des items fréquents les plus spécifiques sont extraites à l'aide de l'algorithme M^2SP.

Parcourant efficacement l'espace de recherche : Nous proposons d'extraire un ensemble non redondant de motifs séquentiels multidimensionnels. Nous définissons ainsi les *motifs séquentiels multidimensionnels clos*. Extraire de tels motifs permet à la fois de disposer d'un ensemble compacte de connaissances sans perte d'information (motif, support) et d'introduire des propriétés supplémentaires d'élagage de l'espace de recherche. Nous proposons deux algorithmes d'extraction de motifs séquentiels clos $CMSP\_Cand$ et $CMSP\_Free$.

Ces deux angles d'attaque débouchent sur des résultats très encourageants et les perspectives associées à chacun de ces points sont nombreuses.

M^2SP permet d'extraire des séquences fréquentes composées uniquement d'items fréquents maximalement spécifiques. Il serait intéressant de continuer à extraire des séquences particulières à partir d'autres types d'items. Les items fréquents clos peuvent être d'excellentes unités de base pour la construction de séquences multidimensionnelles.

Les motifs séquentiels clos permettent d'extraire un ensemble non-redondant de connaissances tout en proposant des propriétés efficaces d'élagage de l'espace de recherche. Dans le contexte des itemsets, il existe de nombreuses représentations condensées autres que les clos. Il serait intéressant de les utiliser pour l'extraction de motifs séquentiels multidimensionnels. Afin d'élaguer l'espace de recherche, nous pouvons aussi nous orienter vers l'extraction de motifs séquentiels multidimensionnels *sous contraintes* (top k).

Extraire de tels motifs permet d'élaguer des branches non prometteuses dans l'espace de recherche à l'aide de calcul de bornes supérieures.

Le nombre de dimensions d'analyse est le paramètre qui, s'il est mal fixé, peut compromettre le passage à l'échelle de l'extraction de motifs séquentiels multidimensionnels. Il serait donc intéressant d'étudier des techniques de réduction du nombre de dimensions d'analyse. Pour cela, nous pouvons utiliser des techniques de recherche de dépendances fonctionnelles afin d'établir qu'une dimension peut être retrouvée à partir d'autres.

La partition de l'ensemble des dimensions $\mathcal{D}$ offre à l'utilisateur une plus grande liberté dans le choix des différents axes (dimensions de référence, d'analyse et temporelles) de l'extraction de motifs séquentiels multidimensionnels. Afin d'améliorer cette interaction, nous pourrions proposer à l'utilisateur d'interdire la présence de valeur joker sur certaines dimensions, de découvrir les motifs contenant uniquement les valeurs a ou b sur une dimensions d'analyse D_a. D'un point de vue théorique, ce ne sont pas des problèmes difficiles à mettre en œuvre, mais d'un point de vue pratique, ils peuvent être d'une grande utilité pour l'utilisateur.

Les données réelles sont souvent multidimensionnelles. Les motifs séquentiels multidimensionnels permettent de prendre en compte cette spécificité et offrent ainsi une meilleure appréhension des données étudiées. Cependant, en pratique, les données sont également agrégées selon différents niveaux de hiérarchies. Par exemple, une cellule d'un cube de données est décrit selon plusieurs niveaux de hiérarchies et est associée à une (ou plusieurs) dimension particulière qui représente le résultat de cette agrégation : *la mesure*. Il est donc primordial de prendre en compte ces spécificités : les hiérarchies et la mesure.

A la fin de cette partie, nous pouvons affirmer que nous avons pris en compte le caractère multidimensionnel des données grâce aux motifs séquentiels multidimensionnels et les algorithmes permettant leur extraction. Nous sommes donc capable de prendre en compte à la fois la multidimensionnalité et la temporalité dans l'extraction de motifs. Toutefois, les bases de données multidimensionnelles contiennent d'autres spécificités (hiérarchies, agrégats) que nous n'avons pas encore abordées. Tel est l'objectif de la partie II où nous nous intéressons à la prise en compte des ces spécificités afin de proposer des algorithmes s'adaptant pleinement aux contextes issus de la vie réelle.

Deuxième partie

Motifs Séquentiels à partir de Bases de Données Multidimensionnelles

La science doit s'accommoder à la nature. La nature ne peut s'accommoder à la science.
Ferdinand Brunot (1921) — *La pensée et la langue*

Dans la partie précédente, nous introduisons les motifs séquentiels multidimensionnels afin de proposer une meilleure appréhension des données examinées. En effet, de nombreux jeux de données sont multidimensionnels ou multi-attributs. Toutefois, les données sont également souvent décrites à l'aide de hiérarchies. Par exemple, on peut facilement établir une hiérarchie sur les produits mis en vente dans un supermarché ou une hiérarchie sur les lieux de ventes. Dans de nombreux contextes, une dimension numérique peut apparaître, représentant le résultat de l'agrégation des faits. Cette dimension est porteuse d'information puisqu'elle « quantifie » les faits. Il est primordial de prendre en compte ces spécificités souvent présentes dans des contextes multidimensionnels (bases de données multidimensionnelles, etc.).

Dans le chapitre 1, nous définissons les motifs séquentiels multidimensionnels h-généralisés où les items sont définis sur plusieurs niveaux de hiérarchies. Nous proposons un algorithme basé sur la même philosophie que M^2SP. L'algorithme M^3SP permet d'extraire des motifs séquentiels h-généralisés à partir des items les plus spécifiques.

Dans le chapitre 2, nous proposons une autre vision de la gestion des hiérarchies en introduisant les séquences convergentes ou divergentes ainsi que les algorithmes permettant leur extraction. Les items de telles séquences se spécialisent (convergent) ou se généralisent (divergent) le long des séquences.

Dans le chapitre 3, nous proposons trois façons différentes de prendre en compte une dimension numérique. Nous introduisons une contrainte d'agrégat sur cette dimension afin de ne prendre en compte que les n-uplets vérifiant la condition. Nous proposons également de discrétiser la ou les dimensions numériques à l'aide de partitions strictes ou floues afin d'intégrer ces dimensions dans l'ensemble des dimensions d'analyse et bénéficier des informations qu'elles contiennent. Enfin, la mesure peut également permettre de calculer le support des séquences multidimensionnelles.

Nous terminons cette partie par une discussions des perspectives associées à ces différentes propositions.

Chapitre 1

M^3SP : Prise En Compte Des Hiérarchies

1.1 Introduction

Dans ce chapitre, nous proposons de prendre en compte les hiérarchies dans l'extraction de motifs séquentiels multidimensionnels. Nous définissons les motifs séquentiels multidimensionnels h-généralisés qui permettent d'extraire des séquences multidimensionnelles définies sur plusieurs niveaux de hiérarchies. L'algorithme M^3SP (Mining Multidimensional and Multiple-level Sequential Patterns) permet d'extraire de tels motifs. Il suit la même philosophie que M^2SP, les motifs séquentiels multidimensionnels h-généralisés sont extraits à partir des items h-généralisés les plus spécifiques. Toutefois, nous montrons qu'une simulation de la gestion des hiérarchies avec M^2SP ne permet pas d'obtenir des résultats aussi performants que M^3SP. Nous avons également mené des expérimentations sur des données synthétiques et sur des jeux de données réels afin d'étudier le comportement de M^3SP en fonction de divers paramètres.

1.2 Prise en compte des hiérarchies dans l'extraction de motifs

Dans cette section, nous présentons les travaux qui proposent de prendre en compte les hiérarchies dans l'extraction de motifs (itemset, motifs séquentiels).

Dans [SA96a], les prémisses de la gestion des hiérarchies dans l'extraction de règles d'association et de motifs séquentiels sont proposés. Les auteurs supposent que les relations hiérarchiques entre les items sont représentées par un ensemble de taxonomies sous forme d'un graphe orienté sans cycle. Ils permettent d'extraire des règles d'association ou des motifs séquentiels suivant plusieurs niveaux de hiérarchies. Ils modifient les transactions en ajoutant pour chaque item l'ensemble de ses ancêtres dans la taxonomie associée. Ensuite, ils génèrent les séquences fréquentes tout en essayant de filtrer au maximun

l'information redondante et en optimisant le processus à l'aide de plusieurs propriétés. Cette approche peut être difficilement adaptée dans un contexte multidimensionnel. En effet, pour chaque transaction, ajouter sur chaque dimension la liste des ancêtres d'un item dans la taxonomie est impensable. Cela reviendrait, dans le pire des cas, à multiplier la taille de la base de données que l'on souhaite étudier par la profondeur maximale d'une hiérarchie et ceci pour chaque dimension d'analyse, un parcours sur cette base serait alors beaucoup trop coûteux.

Dans [HF99], l'approche est sensiblement différente. Les auteurs s'attachent à prendre en compte les hiérarchies durant le processus d'extraction de règles d'association. Même si les auteurs se focalisent sur des règles d'association, il est possible d'étendre cette approche au contexte des motifs séquentiels. Les auteurs proposent d'extraire des règles d'association « intra niveau de hiérarchie ». Ainsi, en partant du plus haut niveau de hiérarchie, les auteurs proposent d'extraire des règles d'association pour chaque niveau de hiérarchie tout en abaissant le support lorsque la recherche de règles d'association s'effectue sur un niveau plus spécifique. La recherche de règles d'association intra niveau est itérée jusqu'à ce qu'aucune règle ne puisse être extraite ou jusqu'à ce que le niveau le plus bas de la hiérarchie soit traité. Cette méthode ne permet donc pas d'extraire des règles où des items issus de différents niveaux de hiérarchie cohabiteraient comme par exemple *Vin* et *Boisson Alcoolisée*. De plus, la mise en œuvre d'une telle approche dans un contexte multidimensionnel suscite de nombreuses questions. Dans le cas où plusieurs hiérarchies existent (une par dimension), doit-on se déplacer sur les mêmes niveaux sur les différentes hiérarchies ou combiner ces niveaux ? Ce type d'extraction peut également être coûteux en temps car le processus d'extraction de connaissances est réitéré tant que des connaissances sont extraites (profondeur de la hiérarchie dans le pire des cas). Ceci n'est pas négligeable dans un contexte multidimensionnel.

Dans [YC05], les auteurs proposent d'extraire des séquences au sein de séquences de données multidimensionnelles organisées en différents niveaux de hiérarchie. Toutefois, la multidimensionnalité et les hiérarchies permettent essentiellement, dans cette approche, de représenter le temps de manière très fine. En effet, les dimensions entretiennent un très fort lien hiérarchique : un utilisateur visite une *page web* durant une *session* pour un *jour* donné. Ainsi, une seule hiérarchie est introduite, et chaque dimension représente un niveau de cette hiérarchie simple. Cette approche ne peut donc pas s'appliquer dans un contexte plus général où il n'existe pas de lien hiérarchique entre les dimensions.

Base de données « exemple »

Pour illustrer les différents concepts et définitions de notre proposition, nous introduisons la table exemple (Tab. 1.1) qui décrit les achats de produits (dimension $Produit$) dans différents lieux (dimension $Lieu$) pour des clients donnés (C_{ID}). Les éléments des dimensions $Produit$ et $Lieu$ sont respectivement décrits par les hiérarchies illustrées par les figures 1.2 et 1.1.

D (Date)	B (C_{ID})	Pl (Lieu)	P (Produit)
1	1	Allemagne	Bière
1	1	Allemagne	Cacahuètes
2	1	Allemagne	Aspirine
3	1	Allemagne	Chocolat
4	1	Allemagne	Smecta
1	2	France	Soda
2	2	France	Vin
2	2	France	Cacahuètes
3	2	France	Aspirine
1	3	UK	Whisky
1	3	UK	Cacahuètes
2	3	UK	Aspirine
1	4	LA	Chocolat
2	4	LA	Smecta
3	4	NY	Soda
4	4	NY	Soda

TAB. 1.1 – Table exemple DB

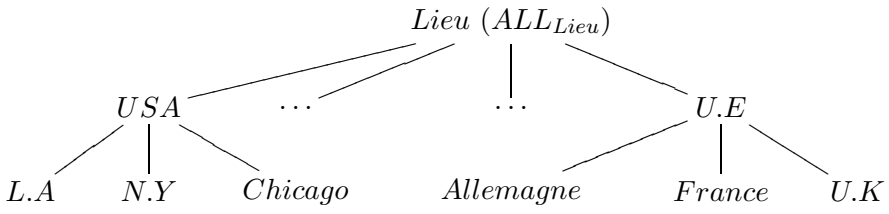


FIG. 1.1 – Hiérarchie sur la dimension $Lieux$

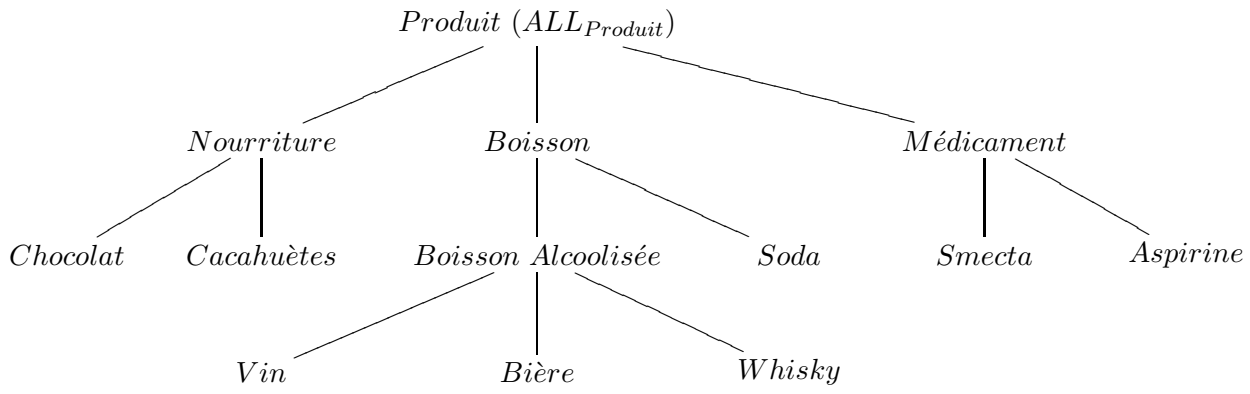


FIG. 1.2 – Hiérarchie sur la dimension $Produits$

1.3 Motifs séquentiels multidimensionnels h-généralisés

Dans cette section, nous définissons le problème de l'extraction de motifs séquentiels multidimensionnels avec prise en compte des hiérarchies. Il est donc nécessaire de présenter comment les hiérarchies peuvent être prises en compte sur les dimensions d'analyse. Nous définissons ainsi, notre modèle de données ainsi que les concepts récurrents d'item, itemset et séquences h-généralisés. Enfin, nous définissons le support des séquences h-généralisées.

1.3.1 Contexte et définitions préliminaires

Soit $\mathcal{D} = \{D_1, D_2, \ldots, D_n\}$, un ensemble de dimensions où chaque dimension D_i est définie sur un domaine potentiellement infini de valeurs, noté $dom(D_i)$. Pour chacune de ces dimensions, nous supposons que $dom(D_i)$ contient une valeur spécifique notée ALL_i.

Afin de prendre en compte le fait que les items puissent être exprimés par rapport à différents niveaux de granularité, nous supposons que chaque dimension D_i est associée à une hiérarchie notée H_i. Chaque hiérarchie H_i est un *arbre* dont les nœuds sont des éléments de $dom(D_i)$ et la racine est ALL_i. Comme souvent, les arêtes d'un tel arbre H_i peuvent être vues comme des relations de type *is-a*. La relation de spécialisation (respectivement la relation de généralisation) correspond à un chemin « top-down » (respectivement « bottom-up ») dans H_i. Un chemin connecte deux nœuds lors d'un parcours de H_i de la racine vers les feuilles (respectivement des feuilles vers la racine).

Lorsqu'il n'y a pas de hiérarchie définie sur une dimension D_i, nous considérons H_i comme un arbre équilibré dont la racine est ALL_i et les feuilles sont tous les éléments de $dom(D_i) \setminus \{ALL_i\}$. La figure 1.3 illustre une telle hiérarchie : pour la dimension X qui n'a pas de hiérarchie définie explicitement, on associe la hiérarchie où la racine est ALL_X et les feuilles sont tous les x_i qui appartiennent à $dom(X)$.

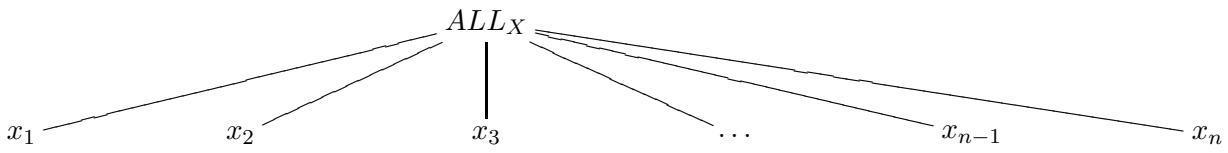


FIG. 1.3 – Hiérarchie H_X « définie arbitrairement » sur D_X

Une table de faits T sur un univers $\mathcal{D}$ est un ensemble fini de n-uplets $t = (d_1, \ldots, d_n)$ où pour chaque $i = 1 \ldots n$, d_i est un élément de $dom(D_i)$ qui est une *feuille* dans la hiérarchie associée H_i. En d'autres mots, nous supposons que la table de faits T contient seulement les valeurs les plus spécifiques par rapport à toutes les hiérarchies associées. En effet, eau ou vin apparaissent sur le ticket de caisse et non $boisson\ alcoolisée$ ou $boisson$.

Etant donnés un élément x de $Dom(D_i)$ et la hiérarchie associée H_i, nous introduisons les notations suivantes :

- $down(x)$ et $up(x)$ représentent respectivement l'ensemble des *spécialisations directes* et le singleton contenant la seule *généralisation directe* de x. Plus précisément, si x n'est pas une feuille dans H_i, alors $down(x)$ est l'ensemble de tous les $y \in Dom(D_i)$ tel que H_i contienne une arête de x à y ; autrement, $down(x)$ est égal à l'ensemble vide.
- Nous notons $x^{\uparrow}$ (respectivement $x^{\downarrow}$) l'ensemble contenant x ainsi que toutes les généralisations (respectivement spécialisations) de x dans H_i. Ainsi, $\forall i = 1 \ldots n, ALL_i^{\uparrow} = \{ALL_i\}, ALL_i^{\downarrow} = Dom(D_i)$, si x est une feuille de H_i alors $ALL_i \in x^{\uparrow}$ et $x^{\downarrow} = \{x\}$. Pour tout x, on a également $down(x) \subseteq x^{\downarrow}, up(x) \subseteq x^{\uparrow}$ et $\{x\} = x^{\uparrow} \cap x^{\downarrow}$.

Exemple 1.1. Par rapport à la table de faits exemple décrite dans le tableau Tab. 1.1, $dom(Produit) = \{bière, soda, vin, whisky, cacahuètes, chocolat, smecta, aspirine\}$.

La figure 1.2 décrit la hiérarchie $H_{Produit}$ associée à la dimension $Produit$. On peut voir à partir de cette hiérarchie que $down(Boisson) = \{Boisson\ Alcoolisée, Soda\}$ and $up(Boisson) = \{ALL_{Produit}\}$.

De plus, à partir de cette hiérarchie, on peut également remarquer que $Boisson$ est une généralisation de $Soda$ puisque $Boisson \in Soda^{\uparrow}$ et $Soda \in Boisson^{\downarrow}$. Comme $Soda$ est une feuille de $H_{Produit}$, on a $down(Soda) = \emptyset$ et $Soda^{\downarrow} = \{Soda\}$.

Puisqu'aucune hiérarchie n'est associée à la dimension $Date$, la hiérarchie implicite associée H_D est définie comme dans la figure 1.3.

Par rapport à la table de faits exemple, nous choisissons la partition de $\mathcal{D}$ suivante :

- $D_A = \{Lieu, Produit\}$
- $D_R = \{C_{ID}\}$
- $D_T = \{Date\}$

– $D_I = \emptyset$

Etant donnée cette partition, on peut identifier 4 séquences de données différentes identifiées par les blocs B_1, B_2, B_3 et B_4. Par exemple, la séquence de données identifiées par B_3 est $\langle\{(UK, Whisky)$,-$(UK, Cacahuètes)\}\{(UK, Aspirine)\}\rangle$. Nous pouvons remarquer que de telles séquences sont décrites avec les valeurs d'agrégation les plus fines sur chaque dimension d'analyse.

Notre but est de découvrir des motifs séquentiels multidimensionnels dans l'ensemble des séquences de données en prenant en compte les hiérarchies, c'est-à-dire extraire des motifs où différents niveaux de hiérarchie sont présents au sein de ces motifs. Il est donc nécessaire de (re-)définir les notions d'item, itemset, et séquence dans un tel contexte.

1.3.2 Item et itemset multidimensionnels h-généralisés

Etant donnée une table de faits DB définie sur un ensemble $\mathcal{D}$ de n dimensions $D_1, D_2, \ldots, D_n$ sur lesquelles des hiérarchies sont définies, nous considérons, comme mentionné précédemment, une partition de $\mathcal{D}$ en quatre sous-ensembles D_A, D_R, D_T et D_I. Nous supposons que $|D_A| = m$ et $m < n - 2$. A partir de ces différents paramètres, nous pouvons définir les concepts fondamentaux d'item et itemset multidimensionnels h-généralisés.

Définition 1.1 (Item multidimensionnel h-généralisé). Un item multidimensionnel h-généralisé $a = (d_1, \ldots, d_m)$ est un m-uplet défini sur D_A, c'est-à-dire $\forall i = 1 \ldots m, \quad D_i \in D_A$, $d_i \in dom(D_i)$ et $\exists j \in [1, \ldots, m]$ tel que $d_j \neq ALL_j$.

Il est important de remarquer que les items multidimensionnels h-généralisés peuvent être définis, contrairement aux faits de la base, avec n'importe quelle valeur sur les hiérarchies associées aux dimensions d'analyse. Ainsi, par rapport à notre table de faits exemple, $(Boisson, USA)$ et $(Boisson\ Alcoolisée, ALL_{Lieu})$ sont des items multidimensionnels h-généralisés.

La dernière condition de la définition, qui réclame l'existence d'au moins une dimension d'analyse instanciée avec une valeur différente de la valeur la plus générale ALL_j permet de ne pas considérer le m-uplet $(ALL_{Lieu}, ALL_{Product})$ qui n'apporte aucune information.

Cette définition est différente de celle des items multidimensionnels de la page 44. En effet, dans cette définition, les items peuvent être définis selon n'importe quel niveau de hiérarchie alors que dans la définition 1.3 de la page 44, les items sont définis qu'avec les niveaux les plus fins des hiérarchies associées ou avec * (ALL).

Puisque les items multidimensionnels h-généralisés sont définis sur plusieurs niveaux de hiérarchies, il est possible d'établir des comparaisons entre eux à partir d'une relation de spécialisation définie de la façon suivante :

Définition 1.2 (Relation de spécialisation). Soient $a = (d_1, \ldots, d_m)$ et $a' = (d'_1, \ldots, d'_m)$ deux items multidimensionnels h-généralisés. On dit que a est plus spécifique que a' (noté $a \subseteq a'$), si $\forall i = 1, \ldots, m$, $d_i \in {d'_i}^{\downarrow}$.

Exemple 1.2. Par rapport à la table de faits exemple, nous avons :

- $(USA, Soda) \subseteq (USA, Boisson)$, car $USA \in USA^{\downarrow}$ et $Soda \in Boisson^{\downarrow}$.
- $(France, Vin) \subseteq (UE, Boisson\ alcoolisée)$, car $France \in UE^{\downarrow}$ and $Vin \in Boisson\ alcoolisée^{\downarrow}$.

Toutefois, certains items peuvent ne pas être comparables entre eux. Ainsi $(France, Vin)$ et $(USA, Soda)$ ne sont pas comparables par rapport aux hiérarchies associées : nous n'avons ni $France \in USA^{\uparrow}$ ni $France \in USA^{\downarrow}$ sur H_{Lieu}.

Il est facile de voir que la relation $\subseteq$ définit un ordre partiel. En d'autres mots, la relation $\subseteq$ définie sur $Dom(D_A)$ est réflexive, antisymétrique et transitive. L'ensemble de tous les items multidimensionnels h-généralisés forme un treillis. En effet, pour tout couple d'items multidimensionnels h-généralisés, il existe une unique borne supérieure et unique borne inférieure.

Afin d'éviter les redondances dans les motifs extraits, deux items comparables ne peuvent pas appartenir au même itemset. Ainsi, comme indiqué dans la définition suivante, deux items peuvent appartenir au même itemset si et s'ils ne sont pas comparables par rapport à l'ordre partiel $\subseteq$.

Définition 1.3 (Itemset multidimensionnel h-généralisé)**.** Un *itemset multidimensionnel h-généralisé* $i = \{a_1, \ldots, a_k\}$ est un ensemble non-vide d'items multidimensionnels h-généralisés tel que $\forall i \in 1 \ldots k$, $\forall j \neq i \in 1 \ldots k$, a_i et a_j ne sont pas comparables par rapport à $\subseteq$ ($a_i \not\subseteq a_j$ et $a_j \not\subseteq a_i$).

Par exemple, $\{(France, Vin), (USA, Soda)\}$ est un itemset multidimensionnel h-généralisé alors que $\{(France, Vin), (UE, Boisson)\}$ n'en est pas un puisque les items $(France, Vin)$ et $(UE, Boisson)$ sont comparables ($(France, Vin) \subseteq (UE, Boisson)$).

La notion de séquence multidimensionnelle h-généralisée découle naturellement des notion d'item et d'itemset h-généralisés. Les séquences h-généralisées ainsi que le support de telles séquences sont définis dans la sous-section suivante.

1.3.3 Séquence multidimensionnelle h-généralisée et support

Définition 1.4 (Séquence multidimensionnelle h-généralisée)**.** Une *séquence multidimensionnelle h-généralisée* $\varsigma = \langle i_1, \ldots, i_j \rangle$ est une liste ordonnée non vide d'itemsets.

Par exemple, $\langle \{(France, Vin), (USA, Soda)\}, \{(Allemagne, bière)\} \rangle$ est une séquence h-généralisée.

Puisque notre objectif est d'extraire des motifs séquentiels multidimensionnels définis sur plusieurs niveaux de hiérarchies, il est nécessaire de pouvoir déterminer le support d'une séquence h-généralisée afin de déterminer si celle-ci est fréquente ou non. La définition suivante décrit comment compter le nombre de blocs de B_{DB,D_R} qui *supportent* la séquence.

Définition 1.5 (Support d'une séquence)**.** Un bloc B_r *supporte* une séquence $\varsigma = \langle i_1, \ldots, i_l \rangle$ si il existe $d_1, \ldots, d_l$ dans $dom(D_T)$ tels que :

1. $d_1 < \ldots < d_l$

2. Pour chaque $j = 1, \ldots, l$ et chaque item α de i_j, B_r contient un n-uplet $t = (d_j, a)$ tel que $a \subseteq \alpha$.

Le *support* de ς correspond au nombre de blocs de B_{DB,D_R} qui supportent la séquence ς (le support relatif étant le pourcentage de blocs de B_{DB,D_R} qui supportent la séquence).

En s'appuyant sur la définition 1.5, nous pouvons définir formellement le problème d'extraction de motifs séquentiels h-généralisés :

Définition 1.6 (Extraction de motifs séquentiels h-généralisés)**.** Etant donné un seuil de support minimun $minsupp$, le but de l'extraction des motifs séquentiels multidimensionnels h-généralisés est d'extraire l'ensemble complet des séquences h-généralisées dont le support est supérieur ou égal à $minsupp$.

L'exemple suivant illustre les définitions précédentes dans le contexte de notre table de faits exemple illustrée Tab 1.1.

Exemple 1.3.
Considérons la séquence $\varsigma = \langle\{(UE, Boisson\ Alcoolisée), (UE, Cacahuètes)\}, \{(EU, Aspirine)\}\rangle$ et le seuil de support relatif minimum $minsupp = \frac{1}{2}$. En considérant tous les blocs B_1, B_2, B_3 et B_4 de $B_{DB,C_{ID}}$, nous avons :

1. B_1. Par rapport aux hiérarchies $H_{Produit}$ et H_{Lieu}, nous avons $Allemagne \in UE^{\downarrow}$, $UK \in UE^{\downarrow}$ et $bière \in$ Boisson Alcoolisée$^{\downarrow}$. Ainsi, $(Allemagne,$ Bière$) \subseteq (UE,$ Boisson Alcoolisée$)$, $(Allemagne,$ cacahuètes$) \subseteq (UE,$ cacahuètes$)$ et $(Allemagne, Aspirine) \subseteq (EU, Aspirine)$. Comme les deux items du premier itemset sont découverts à une date postérieure à la découverte du second itemset, la relation d'ordre est bien respectée. Le bloc B_1 supporte la séquence ς.
2. B_2. Comme $France \in EU^{\downarrow}$ et $Vin \in$ Boisson Alcoolisée$^{\downarrow}$, la séquence ς est également supportée par le bloc B_2.
3. B_3. On est dans un cas similaire aux deux précédents puisque UK appartient à $UE^{\downarrow}$ et $whisky$ appartient à Boisson Alcoolisée$^{\downarrow}$. Le bloc B_3 supporte donc également la séquence ς.
4. B_4. Ce bloc ne supporte pas la séquence ς car aucune valeur de $EU^{\downarrow}$ ne peut être trouvée dans ce bloc.

Etant donné que $|B_{DB,C_{ID}}| = 4$, le support de la séquence ς est égal à $\frac{3}{4}$. La séquence est donc fréquente. ς est un motif séquentiel multidimensionnel h-généralisé.

Dans le but de définir l'inclusion de séquence dans ce contexte, nous devons d'abord définir quand un itemset est un sous-ensemble d'un autre.

Définition 1.7 (Inclusion d'itemsets)**.** Soient deux itemsets i et i', on dit que i est inclus dans i', noté $i \preceq i'$, si pour chaque item a de i, il existe un item a' dans i' tel que $a \subseteq a'$.

Par exemple, on a $\{(France, Vin)\} \preceq \{(EU, Vin)(EU, Soda)\}$ et $\{(USA, Soda)\} \not\preceq \{(NY, Soda),(NY, Chocolat)\}$.

L'ordre partiel $\preceq$ peut facilement être étendu aux séquences afin de définir l'inclusion de séquences de la façon suivante :

Définition 1.8 (Inclusion de séquences). Soient deux séquences h-généralisées $\varsigma = \langle i_1, \ldots, i_l \rangle$ et $\varsigma' = \langle i'_1, \ldots, i'_{l'} \rangle$. On dit que ς est une sous-séquence de ς' (noté $\varsigma \preceq \varsigma'$) s'il existe des entiers $1 \leq j_1 < j_2 < \ldots < j_l \leq l'$ tels que $i_1 \preceq i'_{j_1}, \ldots, i_l \preceq i'_{j_l}$.

Les points suivants sont des exemples d'inclusion de séquences h-généralisées.

- La séquence $\langle \{(France, Vin)\}, \{(Allemagne, bière)\} \rangle$ est une sous-séquence de $\langle \{(France, Vin),(USA, Soda)\}, \{(Allemagne, bière)\} \rangle$.
- La séquence $\langle \{(France, Vin)\}, \{(Allemagne, bière)\} \rangle$ est une sous-séquence de $\langle \{(Paris, \text{Boisson Alcoolisée}),(USA, boisson)\}, \{(UE, \text{Boisson Alcoolisée})\} \rangle$.

Par contre, la séquence $\langle \{(UE, Vin)\}, \{(Allemagne, bière)\} \rangle$ n'est pas une sous-séquence de la séquence $\langle \{(France, Vin)(USA, Soda)\}, \{(Allemagne, bière)\} \rangle$ parce que $(UE, Vin) \not\subseteq (France, Vin)$.

Nous pouvons adapter la définition 3.2 de la page 69 afin prendre de en compte les hiérarchies.

Définition 1.9 (Spécialisation/Généralisation). Un séquence multidimensionnelle h-généralisée $\alpha = \langle a_1, a_2, \ldots, a_l \rangle$ est plus générale que $\beta = \langle b_1, b_2, \ldots, b_{l'} \rangle$ $(l \leq l')$ (et β plus spécifique que α) s'il existe des entiers $1 \leq j_1 \leq j_2 \leq j_l \leq l'$ tels que $b_{j_1} \subseteq a_1, b_{j_2} \subset a_2, \ldots, b_{j_l} \subseteq a_l$.

On note $\alpha \prec_S \beta$.

La séquence $s = \langle \{(UE, Vin)(NY, Soda)\}, \{(Allemagne, bière), (France, Bière)\} \rangle$ est plus spécifique que la séquence $s' = \langle \{(France, Vin)(USA, Soda)\}, \{(Allemagne, bière)\} \rangle$ $(s' \prec_S s)$.

1.4 *M^{3}SP* : Extraction de motifs séquentiels h-généralisés

Dans cette section, nous donnons des propriétés fondamentales sur les motifs séquentiels multidimensionnels h-généralisés, et nous détaillons les algorithmes permettant l'extraction de tels motifs. Ces algorithmes sont basés sur les 2 étapes suivantes :

1. *Etape 1 :* Tout d'abord, les séquences fréquentes les plus spécifiques contenant un seul item sont recherchées. Pour extraire ces items, nous nous basons sur une technique inspirée de l'algorithme de construction d'iceberg (cube dont les cellules respectent une condition d'agrégats) BUC [BR99].
2. *Etape 2 :* Enfin, toutes les séquences fréquentes qui peuvent être construites à partir de items les plus spécifiques sont extraites à l'aide de l'algorithme SPADE [Zak01].

L'adéquation et la complétude de ces deux étapes sont basées sur les propriétés suivantes :

1.4.1 Propriétés

Tout d'abord, notons que les deux étapes décrites précédemment sont basées sur les propriétés d'antimonotonie suivantes :

En considérant que tout item multidimensionnel h-généralisé a peut être vu comme une séquence $\langle\{a\}\rangle$, nous avons la proposition suivante :

Proposition 3. Soient deux items multidimensionnels a et a', si $a \subseteq a'$ alors $support(\langle\{a\}\rangle) \leq support(\langle\{a'\}\rangle)$.

Démonstration. $a \subseteq a'$ signifie que a est plus spécifique que a'.

D'après la définition 1.5, si un bloc supporte la séquence réduite à l'item a, alors ce bloc supporte également $\langle a'\rangle$.

Plus généralement, soient $\mathcal{B}_{\langle a\rangle}$ l'ensemble des blocs de B_{DB,D_R} qui supportent la séquence $\langle a\rangle$ et $\mathcal{B}_{\langle a'\rangle}$ l'ensemble des blocs qui supportent la séquence $\langle a'\rangle$, alors $\mathcal{B}_{\langle a\rangle} \subseteq \mathcal{B}_{\langle a'\rangle}$. Ce qui implique $support(\langle\{a\}\rangle) \leq support(\langle\{a'\}\rangle)$.

□

Maintenant, généralisons la proposition 3 définie ci-dessus, la propriété suivante traite de l'antimonotonie du support des séquences par rapport à la relation d'inclusion $\preceq_S$.

Proposition 4. Soient deux séquences multidimensionnelles h-généralisées ς et ς', si $\varsigma \preceq_S \varsigma'$ alors $support(\varsigma') \leq support(\varsigma)$.

Démonstration. La démonstration est similaire à la démonstration de la propriété 3. Il suffit de montrer que si un bloc supporte ς' alors il supporte ς. □

Notons que dans le cas où les séquences ne contiennent que les items maximalement spécifiques, l'antimonotonie du support fonctionne avec la relation d'inclusion $\prec$ (si $\varsigma \preceq \varsigma'$ alors $support(\varsigma') \leq support(\varsigma)$) puisque l'inclusion des séquences ne peut être due qu'à « la longueur des séquences » (séquences contenant plus d'items) et non à la présence d'items plus généraux.

Les deux propositions précédentes nous permettent d'appliquer le paradigme *a priori* en garantissant la complétude de l'extraction.

Nous passons maintenant aux propriétés requises pour mettre en œuvre efficacement l'étape 1, c'est-à-dire comment extraire toutes les séquences les plus spécifiques contenant un seul item h-généralisé.

Tout d'abord, il est important de remarquer que l'ensemble partiellement ordonné $(Dom(D_A), \subseteq)$ est un treillis. Plus formellement, nous devons considérer un élément supplémentaire, noté $\perp$ qui est plus spécifique que n'importe quel autre item multidimensionnel de D_A.

Ainsi, nous considérons que pour chaque $i = 1, \ldots, m$, l'élément $\emptyset$ est ajouté à $Dom(D_i)$ et ensuite, pour chaque $i = 1, \ldots, m$, et chaque d_i in $Dom(D_i)$ qui est une feuille dans la hiérarchie H_i, $\emptyset$

est l'unique descendant direct de d_i. Notons que ceci est consistant avec les notations précédemment introduites $down(d_i) = \emptyset$. $\perp$ est alors défini comme étant le m-uplet dont tous les éléments sont égaux à $\emptyset$. La relation $\subseteq$ peut donc être étendue à $Dom(D_A) \cup \{\perp\}$ de cette façon afin d'avoir pour chaque item a, $\perp \subseteq a$.

Nous pouvons maintenant définir une borne supérieure et une borne inférieure à tout couple du treillis. Etant donnés, deux items multidimensionnels h-généralisés $a = (d_1, \ldots, d_m$ et $a' = (d'_1, \ldots, d'_m)$, le plus petit majorant et le plus grand minorant, notés respectivement $lub(a, a')$ et $glb(a, a')$ sont définis de la façon suivante :

- $lub(a, a') = (u_1, \ldots, u_m)$ où, $\forall i = 1, \ldots m$, u_i est la valeur la plus spécifique dans $d_i^{\uparrow} \cap (d'_i)^{\uparrow}$.
- $glb(a, a') = (g_1, \ldots, g_m)$ où, $\forall i = 1, \ldots m$, g_i est la valeur la plus générale dans $d_i^{\downarrow} \cap (d'_i)^{\downarrow}$.

Dans nos algorithmes, nous ignorons l'item $\perp$ qui n'a qu'un intérêt théorique. L'étape 1 est effectuée à l'aide d'un parcours en profondeur du treillis $(Dom(D_A), \subseteq)$, en partant de l'élément le plus général $(ALL_1, ALL_2, \ldots, ALL_m)$.

L'algorithme d'extraction des items multidimensionnels h-généralisés suit la même stratégie que BUC [BR99] que nous adaptons dans notre contexte. Ainsi pour chaque item multidimensionnel h-généralisés découverts $a = (d_1, \ldots, d_m)$, nous générons l'ensemble de ses successeurs directs dans $(Dom(D_A), \subseteq$) et, pour chacun d'entre eux nous vérifions leur support sur le sous-ensemble de n-uplets qui sont plus spécifiques que a. Ainsi, plus les items deviennent spécifiques, plus l'ensemble des n-uplets qui le supportent potentiellement se réduit.

Afin de définir l'algorithme correspondant, il est nécessaire de trouver un méthode non redondante pour parcourir efficacement l'espace de recherche, c'est-à-dire de générer une seule fois les successeurs directs d'un item a dans $(Dom(D_A), \subseteq)$.

Soient $succ(a)$ (respectivement $pred(a)$), l'ensemble de tous les successeurs directs (respectivement prédécesseurs directs) d'un item multidimensionnel h-généralisé a, nous avons alors la proposition suivante :

Proposition 5. Pour chaque $a = (d_1, \ldots, d_m)$ de $Dom(D_A)$:

- $succ(a) = \{(d'_1, \ldots, d'_m) \mid (\exists i \in \{1, \ldots, m\})$
$(d'_i \in down(d_i) \wedge (\forall j \neq i)(d'_j = d_j))\}$.
- $pred(a) = \{(d'_1, \ldots, d'_m) \mid (\exists i \in \{1, \ldots, m\})$
$(d'_i \in up(d_i) \wedge (\forall j \neq i)(d'_j = d_j))\}$.

Démonstration. Il faut montrer qu'un $succ(a)$ est le résultat d'une et une seule substitution d'un d_i de a par un élément de $down(d_i)$.

Supposons que $succ(a)$ est le résultat de zéro substitution de d_i de a par un élément de $down(d_i)$. Alors $succ(a) = a$, ce qui n'est pas possible par définition.

Supposons maintenant que $succ(a)$ est le résultat d'au moins deux substitutions de d_i et d_j par $down(d_i)$ et $down(d_j)$ de a où $i \neq j$. Il existe un item plus général que $succ(a)$ et plus spécifique que

a si on fait une substitution d'une des valeurs précédentes (d_i ou d_j) par $up(d_i)$ ou $up(d_j)$ selon le cas. Ainsi $succ(a)$ n'est pas un descendant direct.

$succ(a)$ est donc un descendant direct s'il est le résultat d'une et une seule substitution d'un d_i de a par un élément de $down(d_i)$.

La démonstration pour le prédécesseur direct est similaire. □

Intéressons nous maintenant à la description des différents niveaux du treillis $(Dom(D_A), \subseteq)$. Soit L_0 l'ensemble $L_0 = \{(ALL_1, \ldots, ALL_m)\}$. Pour tout $k > 0$, on considère l'ensemble de tous les items multidimensionnels h-généralisés a tels qu'il existe $k+1$ items multidimensionnels $a_0, \ldots, a_k$ tels que

1. $a_0 = (ALL_1, \ldots, ALL_m)$ et $a_k = a$
2. $\forall i = 1, \ldots k$, $a_i \in succ(a_{i-1})$.

Afin de décrire l'ensemble L_k, nous introduisons la notation suivante : étant donné un item multidimensionnel $a = (d_1, \ldots, d_m)$, $h(a)$ décrit la somme des niveaux de tous les d_i ($i = 1, \ldots, m$) dans la hiérarchie H_i.

Plus formellement, $h(a) = \sum_{i=1}^{i=m} h_i(d_i)$, où $\forall i = 1, \ldots, m$, $h_i(d_i)$ représente le niveau associé à d_i dans la hiérarchie H_i. Ceci nous amène à la proposition suivante :

Proposition 6. Pour chaque $k \geq 0$, l'ensemble L_k de tous les items multidimensionnels de niveaux k dans le treillis $(Dom(D_A), \subseteq)$ est l'ensemble de tous les items a tels que $h(a) = k$.

Démonstration. Cette propriété se démontre facilement par induction sur k. □

Il est facile de voir à partir de la proposition 6, que le nombre de niveaux λ du treillis $(Dom(D_A), \subseteq)$ est égal à $\sum_{i=1}^{i=m}(h_i)$ où h_i correspond à la profondeur de H_i, c'est-à-dire la longueur maximale d'un chemin allant de la racine ALL_i à une feuille de H_i.

La figure 1.4 représente le treillis $((Dom(D_A), \subseteq)$, des cuboïdes pour l'exemple courant où $D_A = \{Lieu, Produit\}$. Nous ne représentons pas l'élément $\perp$.

Il est primordial de s'assurer que toutes les séquences $\langle\{a\}\rangle$ sont considérés au plus une seule fois. En d'autres mots, nous devons générer tous les items multidimensionnels h-généralisés a potentiellement fréquent une seule fois dans notre algorithme.

Dans ce but, nous supposons que les dimensions de D_A sont ordonnées par rapport à un ordre total ; soit $A_1 < \ldots < A_m$ cet ordre. Etant donné un item multidimensionnels h-généralisés $a = (d_1, \ldots, d_m)$, nous notons $\rho(a)$ l'entier défini de la façon suivante :

- Si $a = (ALL_1, \ldots, ALL_m)$, alors $\rho(a) = 0$
- Autrement, $\rho(a)$ est le *plus petit* entier dans $\{1, \ldots, m\}$ tel que $d_{\rho(a)} \neq ALL_{\rho(a)}$ et $\forall j > \rho(a)$, $a_j = ALL_j$.

Ainsi, l'ensemble des *items multidimensionnels h-généralisés générés* à partir d'un item multidimensionnel a se définit de la façon suivante.

Définition 1.10. Pour chaque item multidimensionnel h-généralisé $a = (d_1, \ldots, d_m)$, l'ensemble des *items multidimensionnels h-généralisés générés à partir de* a (noté $gen(a)$) est :

- Si $\rho(a) = 0$ alors $gen(a) = succ(a)$
- Autrement,
  $gen(a) = \{a' = (d'_1, \ldots, d'_m) \mid (\exists i \in \{\rho(a), \ldots, m\})$ où $(d'_i \in down(d_i)) \wedge (\forall j \neq i)(d'_j = d_j)\}$.

La proposition suivante établit que tous les items multidimensionnels h-généralisés de D_A peuvent être générés de manière non-redondante à partir de $(ALL_1, \ldots, ALL_m)$ en utilisant la définition .

Proposition 7. 1. $\forall a \in Dom(D_A)$, $gen(a) \subseteq succ(a)$.

2. $\bigcup_{a \in Dom(D_A)} gen(a) = Dom(D_A) \setminus \{(ALL_1, \ldots, ALL_m)\}$.
3. Quels que soient a et a' appartenant à $Dom(D_A)$, si $a \neq a'$ alors $gen(a) \cap gen(a') = \emptyset$.

Démonstration.

1. Cet item découle de la proposition 5 et de la définition 1.10
2. Soit $a = (d_1, \ldots, d_m) \in Dom(D_A) \setminus \{(ALL_1, \ldots, ALL_m)\}$. Nous avons donc $\rho(a) > 0$ et $d_{\rho(a)} \neq ALL_{\rho(a)}$, ce qui implique que $up(d_{\rho(a)})$ est défini. Soit $a' = (d'_1, \ldots, d'_m)$ tel que $d'_{\rho(a)} = up(d_{\rho(a)})$ et, $\forall i \neq \rho(a)$, $d'_i = d_i$. Alors, il est facile de voir que $a \in gen(a')$ à partir de la définition 1.10.
3. Soient $a = (d_1, \ldots, d_m)$ et $a' = (d'_1, \ldots, d'_m)$ deux items multidimensionnels h-généralisés tels que $a \neq a'$ et $gen(a) \cap gen(a') \neq \emptyset$. Posons $\alpha = (\delta_1 \ldots, \delta_m)$ appartenant à $gen(a) \cap gen(a')$. D'après la définition 1.10 et le point (2) ci dessus, il existe $i \in \{\rho(a), \ldots, m\}$ et $i' \in \{\rho(a'), \ldots, m\}$ tels que $\rho(a) > 0$, $\rho(a') > 0$, et :
   - $\delta_i \in down(d_i)$ et $\forall j \neq i$, $\delta_j = d_j$
   - $\delta_{i'} \in down(d'_{i'})$ $\forall j \neq i'$, $\delta_j = d'_j$

   Supposons que $i \neq i'$, si $i < i'$. Alors, $i' > \rho(a)$, et donc $d_{i'} = ALL_{i'}$. Comme $\delta_{i'} \in down(d'_{i'})$, $\delta_{i'} \neq ALL_{i'}$. Puisque nous avons $\delta_{i'} = d_{i'}$, il y a une contradiction.

   Comme la cas où $i' < i$ est similaire, si $i = i'$ alors $\delta_i \in down(d_i) \cap down(d'_i)$. Puisque H_i est un arbre, si $d_i = d'_i$ alors $a = a'$.

□

Par rapport à l'exemple courant, la figure 1.5 décrit le parcours en profondeur de l'espace de recherche caractérisé par le treillis des cuboïdes de la figure 1.4. La génération des items multidimensionnels h-généralisés s'effectue selon cet ordre et le calcul du support de ces items s'effectuent sur les éléments de la base qui contiennent leur « générateur » (père dans l'arborescence). Ainsi, l'arbre recouvrant de ce treillis est formé par les arêtes dessinées en pointillés.

Nous avons posé les fondements théoriques nous permettant de définir des algorithmes efficaces d'extraction de motifs séquentiels h-généralisés.

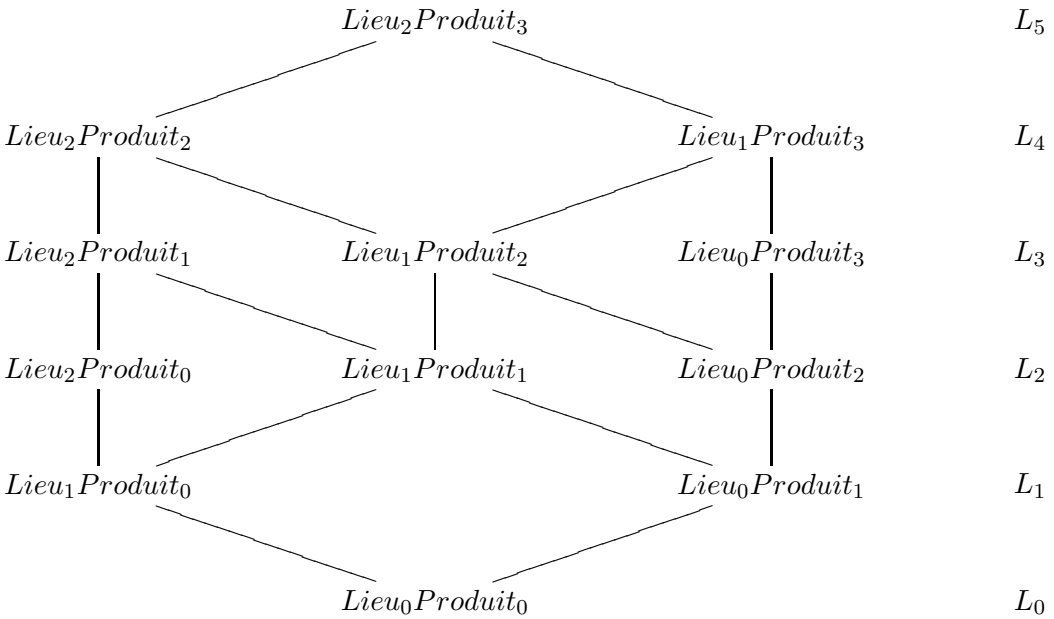


FIG. 1.4 – Treillis des cuboïdes pour l'exemple courant ($D_A = \{Lieu, Produit\}$)

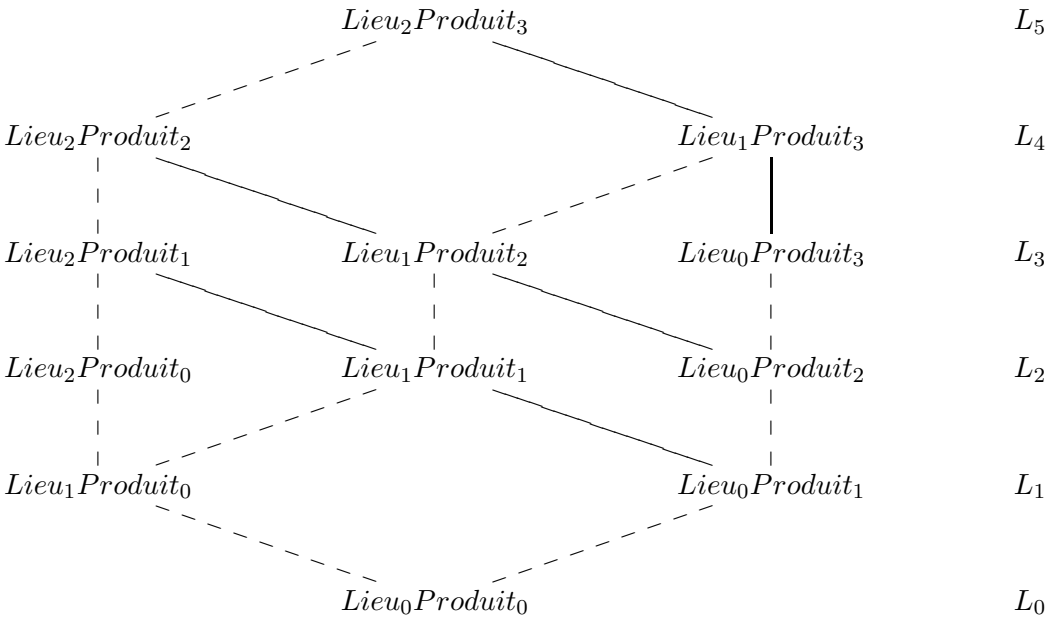


FIG. 1.5 – Parcours arborescent du treillis des cuboïdes pour l'exemple courant ($D_A = \{Lieu, Produit\}$)

1.4.2 Extraction des items multidimensionnels h-généralisés les plus spécifiques

Les algorithmes 12 et 13 décrivent la première étape de l'approche M^3SP, c'est-à-dire l'extraction des items multidimensionnels les plus spécifiques sur une base de données DB étant donnés un seuil de support minimum $minsup$, l'ensemble des dimensions de références D_R, l'ensemble des dimensions d'analyse D_A ainsi que l'ensemble H_A des hiérarchies associées à D_A.

Notre approche diffère du plan d'exécution de BUC sur le premier niveau du treillis des cuboïdes. En effet, BUC considère directement un cuboïde de ce niveau et le traite. Toutefois, dans notre contexte il est possible de découvrir tous les items multidimensionnels du niveau L_1 en une passe sur l'ensemble des données. De plus, cette passe nous permet également d'éliminer les nœuds des hiérarchies non prometteurs et ainsi de limiter la sur-génération d'éléments candidats aux niveaux suivants.

Dans les n-uplets de la base de données, seules les feuilles des hiérarchies associées aux dimensions considérées sont présentes. Ainsi, pour vérifier qu'un n-uplet supporte un item multidimensionnel $a = (d_1, \ldots, d_m)$, il faut contrôler que pour chaque dimension D_i de l'item a, d_i soit un ancêtre de la feuille associée au n-uplet dans la hiérarchie H_i. On effectue donc un parcours des hiérarchies des feuilles vers les fils de la racine pour découvrir les items multidimensionnels fréquents de niveau L_1.

Il est possible d'analyser le *traffic* des éléments des hiérarchies. Le traffic d'un nœud e dans la hiérachie H_i, noté $traffic(e, H_i)$ correspond aux nombres de blocs qui ont permis de parcourir ce nœud lors de la découverte des items de niveau L_1. On peut facilement établir que pour un élément e d'une hiérarchie H_i

$$support(ALL_1, \ldots, ALL_{i-1}, e, ALL_{i+1}, \ldots, ALL_m) \leq traffic(e, H_i)$$

Ainsi si $traffic(e, H_i) < minsup$ alors il n'y a aucune chance de trouver un item spécifique fréquent instancié avec une valeur de $(ALL_1, \ldots, ALL_{i-1}, e, ALL_{i+1}, \ldots, ALL_m)^{\downarrow}$.

En une seule passe sur les données, on extrait donc à la fois les items multidimensionnels de niveaux L_1 et nous détectons les éléments des hiérarchies qui ne devront pas être prises en compte lors des futures générations d'items multidimensionnels candidats.

Pour chaque item multidimensionnel fréquent de niveau L_1 (successeurs directs de $(ALL_1, \ldots, ALL_m)$, l'algorithme 12 appelle l'algorithme 13 qui représente le « cœur » du processus d'extraction des items multidimensionnels les plus spécifiques.

L'algorithme 13 commence par générer les candidats à partir de l'item courant a. Pour cela, la fonction $gen(a)$ (Définition 1.10) est utilisée et les items candidats qui contiennent des valeurs non prometteuses sont filtrés. Le support des items candidats est ensuite calculé à l'aide d'une passe sur la base de données réduite aux n-uplets qui supportent l'item a ($\sigma_a(DB)$).

S'il n'y pas d'items fréquents parmi les items candidats, alors l'item a est potentiellement un item spécifique. Il faut vérifier qu'il n'existe pas déjà d'item plus spécifique dans l'ensemble des items multidimensionnels fréquents les plus spécifiques. S'il existe des items fréquents parmi les candidats, alors l'algorithme est ré-itéré pour tous les items fréquents.

Plus un item a devient spécialisé, plus la base de données, sur laquelle le support des items de $gen(a)$ sera calculé, sera réduite.

Par rapport à l'exemple courant, pour un seuil de support minimum fixé à 2, la figure 1.6 illustre l'extraction des items multidimensionnels h-généralisés les plus spécifiques. L'extraction de ces items est modélisé par un arbre qui a été parcourue en profondeur d'abord. Les feuilles de cet arbre qui sont étiquetées en gras sont les items multidimensionnels h-généralisés les plus spécifiques. Ainsi l'item (ALL,Boisson Alcoolisée) n'est pas un item maximalement spécifique même s'il correspond à une feuille dans l'arbre modélisant la recherche des items les plus spécifiques. En effet, l'item (UE,Boisson Alcoolisée) est plus spécifique.

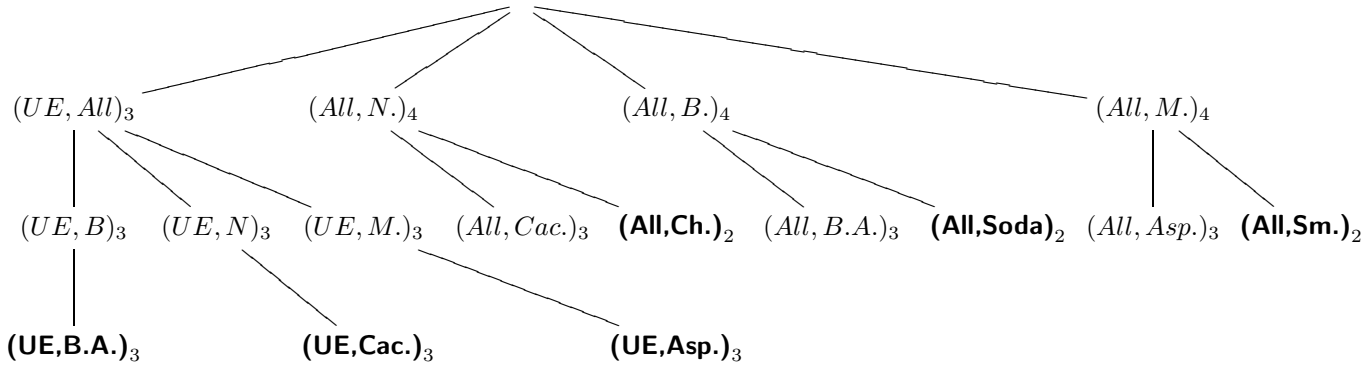


FIG. 1.6 – Arbre d'extraction des items multidimensionnels h-généralisés

1.4.3 Extraction des séquences multidimensionnelles h-généralisées

Les items multidimensionnels h-généralisés les plus spécifiques sont des séquences multidimensionnelles h-généralisées composées de un seul item au sein d'un unique itemset.

Il est possible d'extraire les séquences multidimensionnelles h-généralisée à partir de items les plus spécifiques à partir d'un algorithme d'extraction de motifs séquentiels classiques (mono-dimensionnel). Pour cela, il suffit de transformer les séquences de données définies sur D_A et identifiées par les blocs de B_{DB,D_R} en une séquence de données composées d'items mono-dimensionnels, chaque item mono-dimensionnel identifiant un item multidimensionnel h-généralisé spécifique.

On définit une fonction $numerate()$ qui, à chaque item multidimensionnel h-généralisé spécifique a extrait à l'aide de l'algorithme 12 , associe un unique entier i_a.

Chaque n-uplet de la base de données multidimensionnelles est décomposé en la liste des i_a pour tous les items multidimensionnels a qui sont supportés par ce n-uplet. Sachant qu'on ne considère que

Algorithme 12 : Extraction des items les plus spécifiques

Données : Database DB, minimum support threshold $minsup$, D_A , Hierarchy set H_A, B_{DB,D_R}

Résultat : Set of the most specific items FS

début

/* initialization */

$FS \leftarrow \emptyset$;

$L_1 \leftarrow get\_freq\_items\_from\_L_1(minsup)$;

Tag nodes of hierarchies ;

pour chaque *item* $a \in L_1$ **faire**

call routine FreqItemRec$(a, \sigma_a(DB), minsup, D_A, H_A, B_{DB,D_R}, FS)$;

retourner FS;

fin

Algorithme 13 : Routine FreqItemRec

Données : multidimensional item a, Database DB, minimum support threshold $minsup$, D_A , Hierarchy set H_A, B_{DB,D_R}, FS

Résultat : The current set FS

début

$Cand \leftarrow \{x = (d_1, \ldots, d_m) \ \in gen(a) \mid \nexists i \in 1, \ldots, m \ s.t. \ traffic(d_i, H_i) < minsup\}$;

$Freq \leftarrow \{e \in Cand \mid sup(e) \geq minsup\}$;

si $Freq = \emptyset$ **alors**

si $\nexists a' \in FS \mid a' \subseteq a$ **alors**

$FS \leftarrow FS \cup \{a\}$;

sinon

pour chaque $e \in Freq$ **faire**

/* $\sigma_e(DB)$ represents the elements of each block that support item e */

call $FreqItemRec(e, \sigma_e(DB), minsup, D_A, B_{DB,D_R}, FS)$;

fin

les items les plus spécifiques, il est possible qu'un n-uplet supporte plusieurs items, mais tous ces items sont incomparables entre eux.

Une fois la base transformée, le problème de l'extraction des motifs séquentiels multidimensionnels h-généralisés à partir des items les plus spécifiques peut être vu comme un problème d'extraction de motifs séquentiels classiques. Nous pouvons donc utiliser les algorithmes de la littérature permettant la découverte de motifs séquentiels [Zak01, MCP98, AFGY02, PHMA+04].

Exemple 1.4. Etant donné l'exemple courant centré sur la table de faits Tab. 1.1, les items multidimensionnels les plus spécifiques sont *transformés* en entiers uniques. Le tableau Tab. 1.2 représente la numérotation des items multidimensionnels. A partir de cette numérotation, la base de données est transformée afin d'être directement utilisable par des algorithmes de recherche de motifs séquentiels classiques. Le tableau Tab. 1.3 représente la transformation de la table de faits illustrée dans le tableau Tab. 1.1 en une base de séquences de données mono-dimensionnelles.

Etant donné le seuil de support minimun $minsup$ égal à 2, les motifs séquentiels classiques sont extraits sur la base de séquences (Tab. 1.3). Le tableau Tab. 1.4 énumère les motifs séquentiels maximaux extraits sur la base de séquences Tab. 1.3, ainsi que les motifs séquentiels multidimensionnels h-généralisés maximaux. Pour tous les items des séquences maximales classiques, on applique la fonction $numerate^{-1}()$ afin d'obtenir les items multidimensionnels h-généralisés associés.

Item a	i_a
(U.E, Boisson Alcoolisée)	1
(U.E, Cacahuètes)	2
(U.E, Aspirine)	3
(ALL_{Lieu}, Chocolat)	4
(ALL_{Lieu}, Soda)	5
(ALL_{Lieu}, Smecta)	6

TAB. 1.2 – Numérotation des items les plus spécifiques

B_{ID}	Séquence de données
1	$\langle(1,2),3,4,6\rangle$
2	$\langle 5,(1,2),3\rangle$
3	$\langle(1,2),3\rangle$
4	$\langle 4,6,5,5\rangle$

TAB. 1.3 – Base de données transformée

Motifs Séquentiels Maximaux	Motifs Séquentiels Multidimensionnels Maximaux
$\langle(1,2),3,\rangle$	$\langle$\{(U.E, Boisson Alcoolisée),(U.E, Cacahuètes) \}$\{(U.E, Aspirine)\}\rangle$
$\langle 5\rangle$	$\langle\{(ALL_{Lieu}, Soda)\}\rangle$
$\langle 4,6\rangle$	$\langle\{(ALL_{Lieu}, Chocolat)\}\{(ALL_{Lieu}, Smecta)\}\rangle$

TAB. 1.4 – Les motifs séquentiels multidimensionnels h-généralisés maximaux

1.5 M^3SP Vs M^2SP

Dans cette partie, nous montrons la différence entre l'approche M^3SP définie dans ce chapitre et l'approche M^2SP définie précédemment.

Même si les deux approches suivent la même philosophie (extraction des séquences fréquentes à partir des items les plus spécifiques), elles sont foncièrement différentes. M^2SP peut être vue comme un cas particulier de l'approche M^3SP où toutes les hiérarchies sont de profondeur 1.

M^3SP permet également une gestion plus fine des valeurs jokers. En effet, dans M^2SP, les items multidimensionnels sont instanciés soit par des valeurs fréquemment présentes dans les données (feuilles des hiérarchies), soit par la valeur joker qui est similaire à ALL_i.

Par rapport à l'exemple courant, la figure 1.7 représente les cuboïdes qui sont considérés par M^2SP. Les éléments en gras sont les cuboïdes dont les valeurs associés sont soit feuille, soit racine des hiérarchies. Avec l'approche M^2SP sur la base de données exemple, trois séquences maximales sont extraites :

- $(*, Soda)$
- $\langle\{(*, Chocolat)\}\{(*, Smecta)\}\rangle$
- $\langle\{(*, Cacahuetes)\}\{(*, Aspirine)\}\rangle$

Ces séquences décrivent moins bien les données sources que celles extraites avec l'approche M^3SP (Tab. 1.4). En effet, la séquence multidimensionnelle $\langle$\{(U.E, Boisson Alcoolisée),(U.E, Cacahuètes) \}-$\{(U.E, Aspirine)\}\rangle$ est extraite avec M^3SP au lieu de la séquence $\langle\{(*, Cacahuetes)\}\{(*, Aspirine)\}\rangle$.

Les figures 1.8 et 1.9 illustrent la gestion totalement différente des valeurs jokers pour M^2SP et M^3SP. La racine ALL_i d'une hiérarchie H_i représente la valeur joker * sur la dimension D_i. S'il n'existe pas d'instanciation possible avec une feuille de la hiérarchie, alors on passe directement à la racine de la hiérarchie (Fig. 1.8).

La prise en compte des hiérarchies, permet d'extraire des connaissances plus fines. En effet, les hiérarchies proposent plusieurs alternatives par rapport à l'approche M^2SP quand on n'arrive pas à instancier une dimension avec une feuille de la hiérarchie. En effet, on ne passe pas directement de la feuille à la racine, on essaie d'instancier par l'ancêtre le plus spécifique de la feuille (Fig. 1.9).

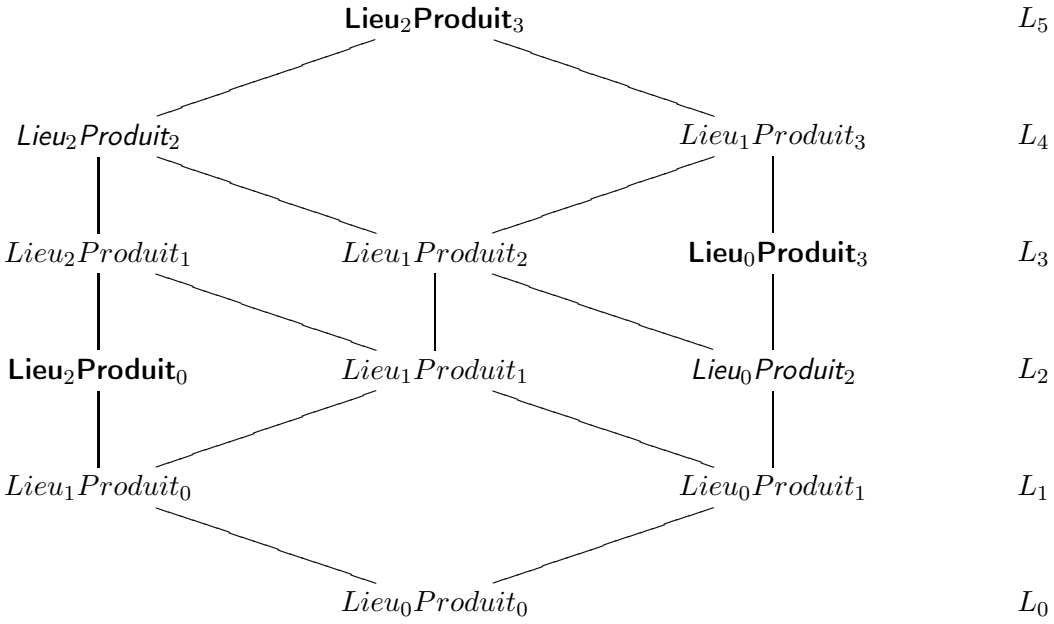


FIG. 1.7 – Treillis des cuboïdes pris en compte avec M^2SP ($D_A = \{Lieu, Produit\}$)

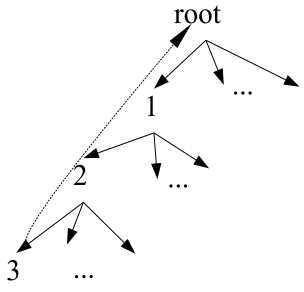


FIG. 1.8 – Gestion de la valeur joker (*)

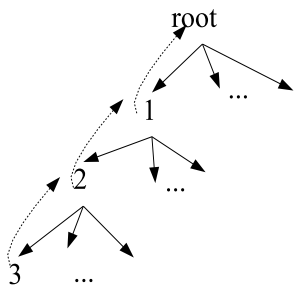


FIG. 1.9 – Gestion des hiérarchies

1.6 Expérimentations

Des expérimentations ont été effectuées afin de montrer la pertinence de M^3SP. Dans ce but, nous montrons tout d'abord l'intérêt d'utiliser directement M^3SP que simuler la gestion des hiérarchies avec M^2SP à l'aide d'une transformation de la base de données. Nous analysons ensuite le comportement de M^3SP par rapport à différents paramètres sur des jeux de données synthétiques. Ces expérimentations visent à caractériser les promesses et les limites de M^3SP en vue d'un passage à l'échelle. Nous finissons par des expérimentations sur des jeux de données réels.

1.6.1 Simulation de la gestion des hiérarchie avec M^2SP

Il est possible de simuler M^3SP avec l'approche M^2SP. Pour cela, il faut transformer la base de données afin de faire explicitement apparaître les ancêtres des feuilles de la hiérarchie dans les données. On aura ainsi une dimension d'analyse par niveau de hiérarchie. Par exemple, le n-uplet (Allemagne,Bière) devient (Allemagne, U.E, ALL_{Lieu},Bière, Boisson Alcoolisée, Boisson, $ALL_{Produit}$). Afin que tous les n-uplets soient décrits sur le même nombre de dimensions, nous supposons que les hiérarchies H_i sont ici des arbres équilibrés. Le nombre de dimensions d'analyse devient donc dans ce contexte $|D_A^{M^2SP}| = \sum_{i=1}^{m} depth(H_i) + 1$.

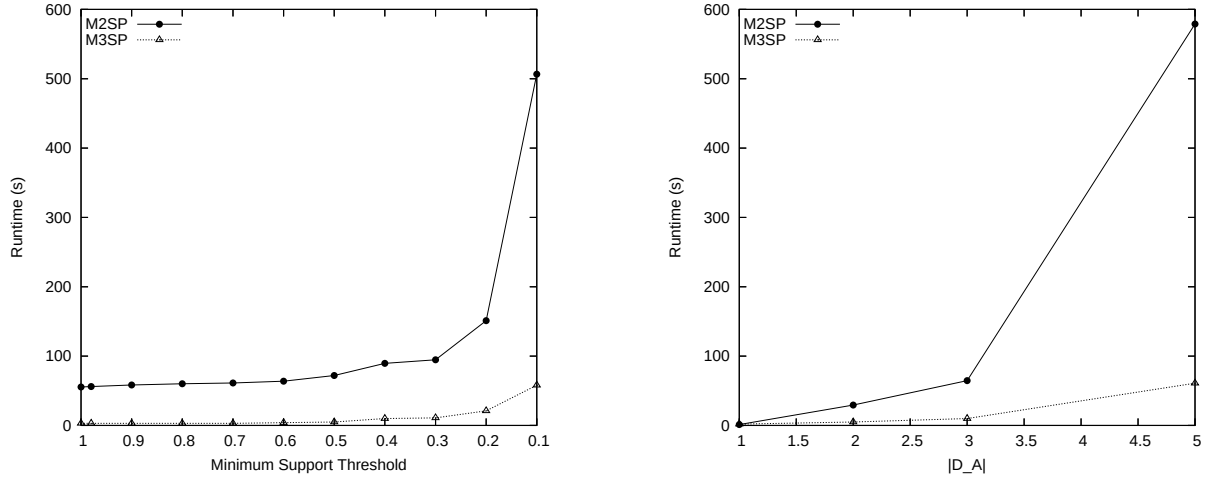
En transformant la base de données de cette manière, il est ensuite possible d'extraire les items fréquents à l'aide l'algorithme M^2SP tout en filtrant les items afin d'extraire les plus spécifiques. Par exemple, si on a (*, U.E, ALL_{Lieu},*, Boisson Alcoolisée, Boisson, $ALL_{Produit}$), cet item et équivalent à l'item h-généralisé (U.E, Boisson Alcoolisée).

Nous montrons dans cette partie qu'une telle simulation n'est pas envisageable en pratique. En effet, il est plus efficace de directement utiliser l'approche M^3SP pour extraire des motifs séquentiels multidimensionnels h-généralisés.

La figure 1.10(a) rapporte le temps d'exécution de M^3SP et de la simulation de la prise en compte des hiérarchies avec M^2SP en fonction du seuil de support minimum considéré. La base de données synthétiques considérée est composée de 50000 blocs et 4 dimensions d'analyse. La profondeur moyenne des hiérarchies est 3. Ainsi la prise en compte des hiérarchies avec M^2SP nécessite la transformation de la base en une nouvelle base comportant 16 dimensions d'analyse. Le coût de cette transformation n'est pas pris en compte dans les temps d'exécution rapportés. Le temps d'exécution de M^2SP est très nettement supérieur à celui de M^3SP. En effet, le temps d'extraction de M^3SP pour $minsupp = 0.1$ est inférieur au temps d'exécution de M^2SP pour $minsupp = 1$.

La figure 1.10(b) décrit le temps d'exécution de ces deux approches en fonction du nombre de dimensions d'analyse considérées ($|B_{DB,D_R}| = 30000$, $minsupp = 0.5$). Simuler la gestion des hiérarchies avec M^2SP est envisageable quand le nombre de dimensions d'analyse est très faible ($|D_A| < 3$). En effet, le temps d'exécution de M^2SP est sensiblement supérieur à celui de M^2SP. De plus, la pente de la courbe de M^2SP augment très rapidement, ce qui compromet le passage à l'échelle d'une telle approche

pour prendre en compte les hiérarchies dans l'extraction de motifs séquentiels multidimensionnels h-généralisés. Ceci est dû au fait que le nombre de dimensions d'analyse réellement considéré est $|D_A^{M^2SP}| = \sum_{i=1}^{m} depth(H_i) + 1$ et $|D_A^{M^2SP}|$ est sensiblement plus grand que $|D_A|$.



(a) Temps d'exécution en fonction du seuil de support minimum

(b) Temps d'exécution en fonction du nombre de dimensions d'analyse

FIG. 1.10 – Comparaison entre M^3SP et M^2SP

Les expérimentations de la figure 1.10 montrent l'intérêt M^3SP. Plus précisément, ces expérimentations montrent la nécessité de prendre en compte les hiérarchies directement dans le processus d'extraction de motifs séquentiels multidimensionnels h-généralisés et par l'intermédiaire de pré-traitement des données et d'application de filtres. Nous nous intéressons maintenant au comportement de M^3SP par rapport à différents paramètres en vue d'un passage à l'échelle de cette approche.

1.6.2 M^3SP

Dans cette partie, nous étudions le comportement de M^3SP à l'aide d'expérimentations menées à la fois sur des jeux de données synthétiques et sur des jeux de données réels. Ces expérimentations ont pour objectif de montrer l'intérêt de M^3SP, notamment le passage à l'échelle de cette approche en fonction de certains paramètre (taille de la base de données, nombre de dimensions d'analyse, spécifités des hiérarchies, etc.).

Jeux de données synthétiques

Les figures 1.11(a) et 1.11(b) décrivent le comportement de M^3SP en fonction de la taille de la base de données considérée. Notons que la taille de la base de données est exprimée en nombre de blocs ($|B_{DB,D_R}|$). Chaque bloc contient plusieurs n-uplets. Ainsi, une base de 500000 blocs correspond à environ 5 million de n-uplets dans les jeux de données considérés. Le temps d'exécution de M^3SP

augmente proportionnellement à la taille de la base de données. Nous remarquons que la découverte de motifs séquentiels multidimensionnels h-généralisés est plus rapide avec des seuils de support minimum élevés ou un faible nombre de dimensions d'analyse. En effet, plus le seuil de support minimum est faible, plus l'espace de recherche est important (plus de motifs fréquents). Plus le nombre de dimensions d'analyse est important, plus grand est le treillis des cuboïdes associés. Par rapport à ces courbes, nous pouvons affirmer que $M3SP$ passe à l'échelle par rapport à la taille de la base de données examinée.

La figure 1.11(c) décrit le temps d'exécution de M^3SP en fonction du seuil de support minimum. Evidemment, le temps d'exécution augmente quand le seuil de support minimum décroît. En effet, moins le seuil de support est contraignant, plus l'espace de recherche est important. Ce phénomène est plus sensible quand le nombre de dimensions d'analyse est important car il y a plus de combinaisons fréquentes à prendre en compte. Ainsi, la pente de la courbe est plus importante, ce qui ne nous permet pas de conclure à un passage à l'échelle total de M^3SP. En effet, cette approche ne passe pas suffisamment à l'échelle lorsque le nombre de dimensions d'analyse est important et le seuil de support est faible. Nous pouvons toutefois estimer que M^3SP passe à l'échelle en fonction du seuil de support minimum lorsque le nombre de dimensions d'analyse n'est pas trop important.

La figure 1.11(d) décrit le temps d'exécution de M^3SP en fonction de la profondeur moyenne des hiérarchie associées aux dimensions d'analyse. Ajouter un niveau de hiérarchie sur une dimension équivaut à spécialiser les données. Par exemple, les *Boissons* deviennent des *S*odas ou des *B*oissons Alcoolisées. Le temps d'exécution de M^3SP augmente lorsque la profondeur moyenne des hiérarchies augmente. En effet, l'espace de recherche devient plus important puisque le treillis des cuboïdes associé devient plus profond. Nous remarquons également que la pente de la courbe devient sensiblement plus forte lorsque la profondeur moyenne des hiérarchies avoisine 10. En pratique, la profondeur moyenne des hiérarchies n'est pas si importante. Nous pouvons donc considérer que M^3SP passe à l'échelle par rapport à ce paramètre.

La figure 1.11(e) rapporte le temps d'exécution de M^3SP en fonction du nombre de dimensions d'analyse considérée. Comme nous l'avons déjà remarqué, plus le nombre de dimensions d'analyse est important, plus le nombre d'items fréquents est important. De plus, la taille de la base de données considérée devient plus importante à cause la taille des n-uplets examinés (plus de dimensions). La valeur du seuil de support minimum accentue l'écart les temps d'exécution de M^3SP en fonction de différents seuils. Par exemple, le temps d'extraction de M^3SP pour un seuil de support égal à 0.3 et $|D_A| = 15$ est similaire au temps nécessaire pour pour un support égal à 0.2 et $|D_A = 10|$. Ce phénomène ne nous permet pas de conclure à un passage à l'échelle total par rapport au nombre de dimensions d'analyse considérées.

La figure 1.11(f) rapporte le temps d'exécution de M^3SP en fonction du degré sortant moyen des hiérarchies. Augmenter le degré sortant d'une hiérarchie équivaut à spécialiser certaines valeurs ce ce niveau (spécialisation au niveau de la fratrie). Par exemple, *Kosovo* peut devenir un frère de *Serbie* dans la hiérarchie alors qu'avant cette spécialisation, un nœud représentait les deux valeurs. Il est très difficile

de prédire le comportement de M^3SP. En effet, supposons que l'item (X, Y) était fréquent avant la spécialisation de la fratrie (X et X'), l'item (X, Y) peut rester fréquent ou devenir non-fréquent. L'item (X', Y) peut aussi apparaître comme fréquent. Si nous analysons les courbes de la figure 1.11(f), le temps d'exécution reste globalement inchangé pour $|D_A| = 3$ et $|D_A| = 5$. Pour $D_A = 10$, le temps d'exécution n'est pas strictement croissant. Nous souhaitons souligner le fait que la taille des hiérarchies devient plus importante et un nombre plus important de combinaisons possibles doit être pris en compte.

Jeux de données réels

Nous avons également effectué des expérimentations sur des jeux de données réels. En effet, des expérimentations ont été menées sur des cubes de données issus de l'activité marketing d'EDF décrit dans l'annexe A. Ces cubes de données décrivent l'activité marketing sur une grande base de données de clients (environ 30 million de clients résidentiels).

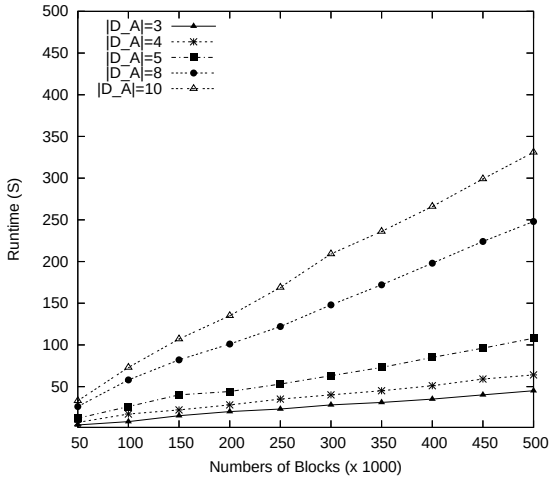
Pour le premier cube de données (Fig. 1.12(a) et Fig. 1.12(b)), $|D_A| = 6$ and $|B_{DB,D_R}| = 8$. Dans le second cube de données (Fig. 1.12(c) et Fig. 1.12(d)), $|D_A| = 5$ et $|B_{DB,D_R}| = 85$.

La figure 1.12(a) décrit le comportement de M^3SP en fonction du seuil de support minimum considéré pour le premier cube de données. Le temps d'exécution augmente lorsque le seuil de support diminue. La figure 1.12(b) qui rapporte le nombre d'items multidimensionnels h-généralisés les plus spécifiques et le nombre total, explique ce phénomène. En effet, le nombre d'items fréquents augmente sensiblement lorsque le support diminue. Les items représentent des séquences de longueur 1. Ceci implique l'augmentation du nombre de séquences extraites. Le temps d'exécution de M^3SP augmente donc. Nous remarquons également que le nombre d'items multidimensionnels h-généralisés les plus spécifiques est considérablement inférieur au nombre total d'items multidimensionnels h-généralisés fréquents, ce qui permet à M^3SP d'éviter le coût combinatoire de l'extraction de toutes les séquences.

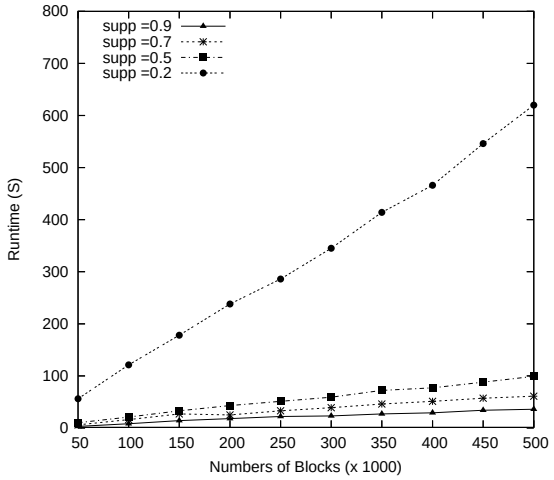
La figure 1.12(c) rapporte le comportement de M^3SP en fonction du seuil de support minimum pour le second cube de données. Ce comportement est similaire à ceux des courbes des figures 1.11(c) et 1.12(a). Nous notons que le nombre d'items fréquents (Fig. 1.12(d)) est relativement important (environ 16000 pour un seuil de support minimum égal à 0.15). La différence entre le nombre d'items les plus spécifiques et le nombre total d'items devient plus importante lorsque le support diminue.

1.7 Discussion

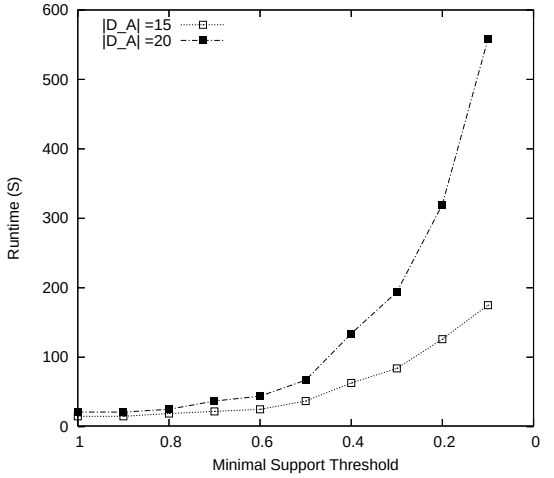
Dans ce chapitre, nous définissons les motifs séquentiels multidimensionnels h-généralisés qui permettent le prise en compte des hiérarchies sur les dimensions d'analyse. La gestion des hiérarchies permet la découverte de séquences multidimensionnelles de la forme $\langle\{(UE, Boisson\ Alcool.), (UE, cacahuètes)\}$-$\{(UE, aspirine)\}\rangle$, définies sur différents niveaux de hiérarchies indiquant que les individus ayant acheté



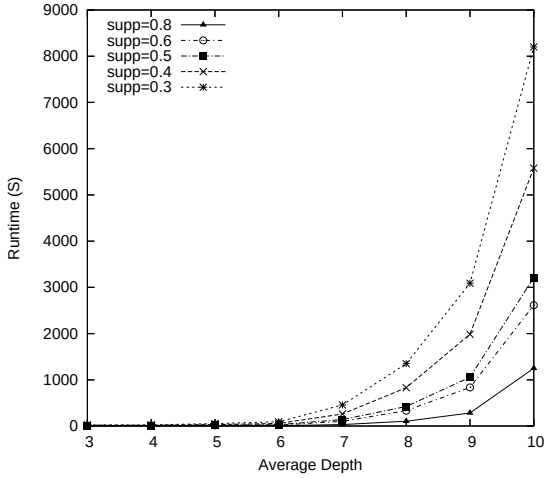
(a) Temps d'exécution en fonction de la taille de la base de données pour différents choix de D_A



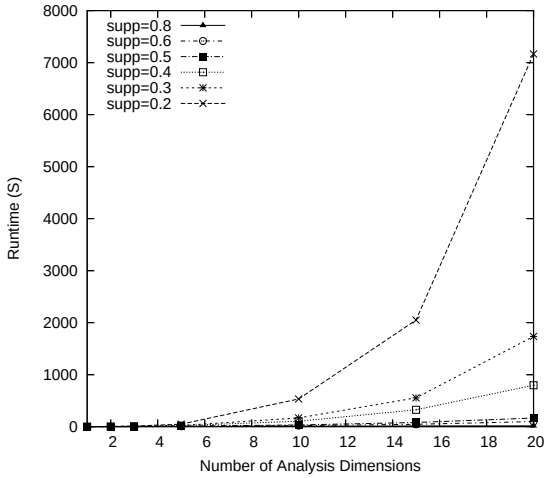
(b) Temps d'exécution en fonction de la taille de la base de données pour différents choix de $minsupp$



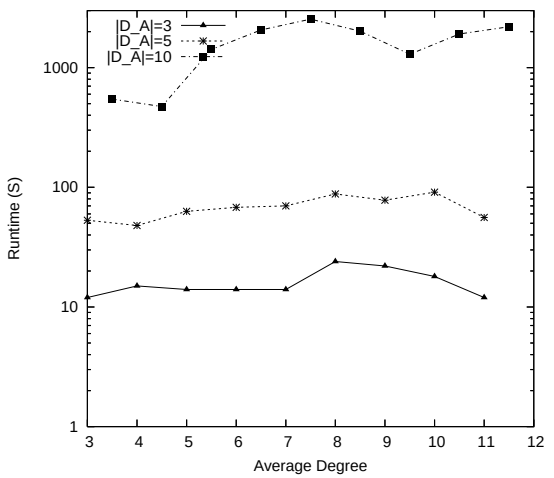
(c) Temps d'exécution en fonction en fonction de $minsupp$ pour différents choix de D_A



(d) Temps d'exécution en fonction de la profondeur moyenne des hiérarchies

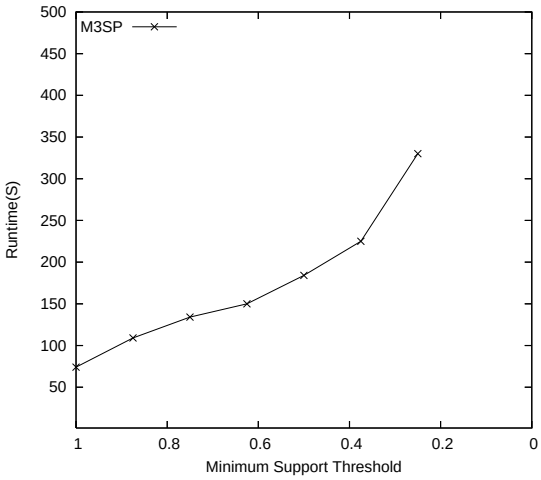


(e) Temps d'exécution en fonction de $|D_A|$

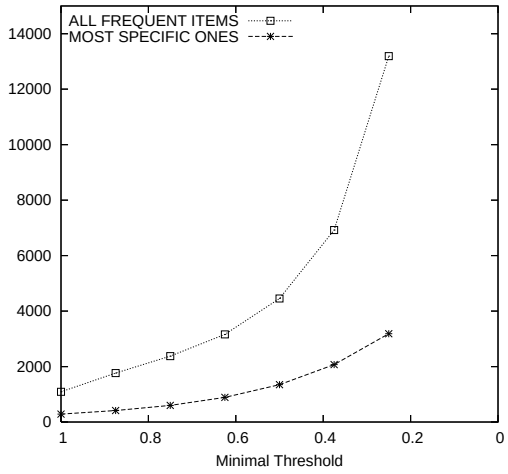


(f) Temps d'exécution en fonction du degré sortant moyen des hiérarchies

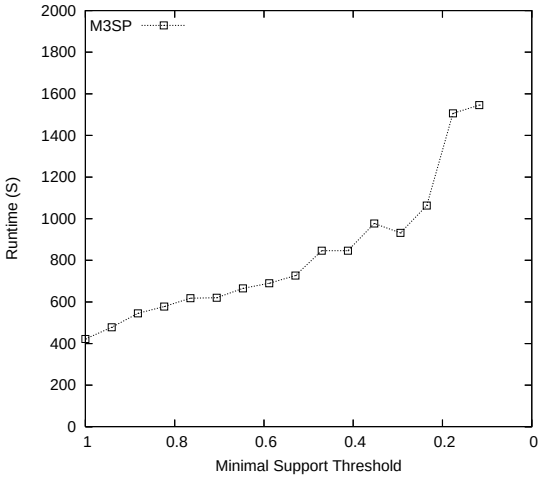
FIG. 1.11 – Expérimentations menées sur des jeux de données synthétiques



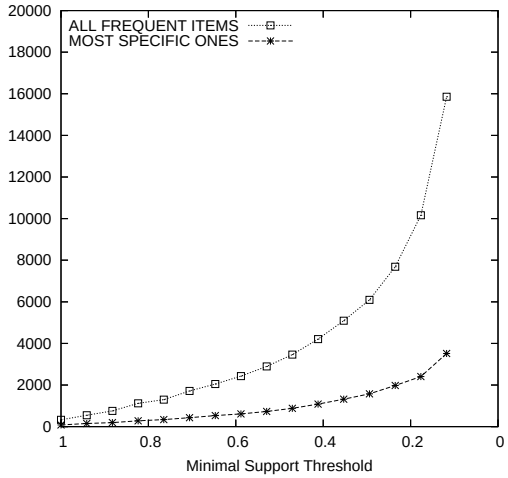
(a) Temps d'exécution en fonction du seuil de support minimum



(b) Nombre d'items multidimensionnel h-généralisés fréquents en fonction du seuil de support minimum



(c) Temps d'exécution en fonction du seuil de support minimum



(d) Nombre d'items multidimensionnel h-généralisés fréquents en fonction du seuil de support minimum

FIG. 1.12 – Expérimentations menées sur des jeux de données réelles

dans l'union européenne des cacahuètes et des boissons alcoolisées en même temps achètent ensuite dans l'union européenne de l'aspirine.

Nous définissons les différents concepts (item, itemset et séquence h-généralisés) et l'algorithme M^3SP permettant la prise en compte des motifs séquentiels multidimensionnels h-généralisés. Des expérimentations menées sur des données réelles et synthétiques montrent l'efficacité de M^3SP. Nous montrons également que la prise en compte des hiérarchies n'est pas un problème trivial. Ainsi, simuler la gestion des hiérarchies à l'aide de l'algorithme M^2SP est beaucoup plus coûteux qu'utiliser directement l'algorithme M^3SP.

Dans notre proposition, nous définissons les hiérarchies comme des arbres. Or en pratique, les hiérarchies sont des généralement des graphes orientés sans cycle (DAG). En effet, les DAGs permettent les hiérarchies multiples. L'utilisation de DAGs dans notre proposition nécessitent quelques ajustements. En effet, dans un DAG, il peut exister deux chemins différents entre la racine et un nœud du graphe. Garantir la non redondance de la génération de candidats devient plus difficile. Toutefois, étant donné que nous utilisons un parcours de l'espace de recherche en *profondeur d'abord*, nous pouvons mettre en œuvre un mécanisme de filtre afin de ne pas générer un nœud plus d'une fois. L'utilisation de hiérarchies décrites sous forme de DAGs peut donc être facilement adaptée à l'algorithme M^3SP.

Cependant, les DAGs sont le support des hiérarchies multiples. Ces hiérarchies multiples proposent différents chemins entre la racine et une feuille. Il est important de remarquer que chaque chemin est associer à une sémantique particulière. L'intégration des hiérarchies sous forme de DAGs doit prendre en compte la sémantique de chaque chemin. Ainsi, privilégier un chemin par rapport à un autre peut permettre d'élaguer plus rapidement l'espace de recherche.

D'autres perspectives peuvent être associées à la proposition développée dans ce chapitre. Elles sont de deux types :

1. Tout d'abord, nous pouvons améliorer le *processus d'extraction* des motifs séquentiels multidimensionnels h-généralisés en introduisant des techniques d'élagages supplémentaires.
2. Ensuite, nous pouvons améliorer également la *qualité* des connaissances extraites. Pour cela, nous pouvons imaginer une gestion modulaire des hiérarchies où certaines dimensions n'auraient pas le même comportement que les autres afin s'adapter aux besoins de l'utilisateur (interdiction de dépasser le niveau de hiérarchie λ sur la dimension ξ, ...). La gestion des hiérarchies peut nous amener à définir une nouvelle méthode automatisée d'aide à la navigation dans les cubes de données.

L'extraction de motifs séquentiels multidimensionnels h-généralisés peut s'appliquer dans des cubes de données et représenter un excellent outil pour le décideur. Toutefois, afin de s'adapter complètement à de tels contextes, il est primordial de prendre en compte une autre spécificité : la présence de valeur agrégées par l'intermédiaire des mesures des cubes de données. C'est l'objet du chapitre 3, mais avant cela, nous allons adopter un autre point de vue sur la prise en compte des hiérarchies dans le chapitre 2.

Chapitre 2

Extraction de Séquences Convergentes et Divergentes

Dans ce chapitre, nous proposons une approche totalement différente d'extraction de motifs séquentiels multidimensionnels h-généralisés. Nous proposons d'extraire des séquences h-généralisées convergentes et divergentes.

La prise en compte des hiérarchies dans l'extraction de motifs séquentiels multidimensionnels a été proposée dans le chapitre précédent via l'algorithme M^3SP qui permet l'extraction de motifs séquentiels multidimensionnels h-généralisés. Néanmoins, cette proposition ne permet pas d'extraire des motifs de la forme : *"Quand les ventes de Perrier augmentent en France, les ventes de boissons non alcoolisées augmentent en Europe le mois suivant"* où les deux items multidimensionnels présents dans la séquence $(Perrier, France)$ et $(Boisson\ non\ Alcoolisée, Europe)$ sont comparables ($(Perrier, France) \subseteq (Boisson\ non\ Alcoolisée, Europe)$).

C'est pourquoi nous proposons d'extraire de tels motifs en prenant en compte l'une des principales singularités du contexte multidimensionnel : les hiérarchies. Nous introduisons les concepts de séquences multidimensionnelles convergentes et divergentes. Ces concepts permettent d'extraire des séquences plus longues en modulant le degré de précision/généralisation le long de celles-ci. Une séquence convergente, qui va du général au particulier, est par exemple, *"Quand les ventes de sodas augmentent aux USA, les ventes de coca augmentent sur la côte ouest ainsi que les ventes de pepsi sur la côte est"* alors qu'une séquence divergente, qui va du particulier au général, est *"Quand les ventes de Perrier augmentent à Nice et que les ventes de coca augmentent à Munich, les ventes de boissons non-alcoolisées augmentent dans l'UE"*.

Ce chapitre s'organise de la façon suivante. Tout d'abord, nous introduisons les définitions des motifs séquentiels multidimensionnels h-généralisés convergents et divergents. Les algorithmes $M2S\_CD$ permettant l'extraction de tels motifs sont basés sur les algorithmes définis précédemment. Des expérimentations effectués sur des jeux de données synthétiques et réels sont décrites et montrent l'intérêt de

cette proposition. Enfin, nous discutons des améliorations possibles de cette proposition dans la dernière section.

2.1 M2S_CD : motifs séquentiels multidimensionnels convergents ou divergents

Dans cette section, nous introduisons un concept original visant à reproduire le raisonnement humain grâce aux hiérarchies. En effet, l'esprit humain raisonne souvent de deux façons différentes et symétriques. La réflexion s'exécute de l'exemple vers la théorie ou de la théorie vers l'exemple. De plus, la précision d'un fait diminue avec le temps. Par exemple, un cuisinier sait ce qu'il a cuisiné au dernier service, il ne sait plus précisément ce qu'il fait la veille, et ne sait plus du tout ce qu'il a cuisiné deux ans avant. Nous essayons donc de reproduire ce type de raisonnement dans les connaissances que nous souhaitons extraire. Nous introduisons donc le concept de *séquence multidimensionnelle convergente ou divergente*. Nous présentons les différentes définitions préliminaires associées aux motifs séquentiels multidimensionnels avec prise en compte des hiérarchies pour ensuite détailler les concepts de motifs convergents et divergents ainsi que les algorithmes associés.

Base de données « exemple »

Pour illustrer les différents concepts et définitions, nous proposons la base de données exemple illustrée dans le tableau Tab. 2.1 qui décrit les ventes réalisées dans différentes villes par différentes chaînes de magasins. Deux dimensions sont pourvues de hiérarchies : les villes et les boissons respectivement indiquées par les figures 2.1 et 2.2.

Par rapport à la partition de l'ensemble des dimensions, nous choisissons les valeurs suivantes :

- $D_R = \{Enseigne\}$
- $D_T = \{Date\}$
- $D_A = \{Lieu, Boisson\}$: la base de données exemple est déjà exprimée sous la forme d'items multidimensionnels.
- $D_I = \emptyset$

Séquence multidimensionnelle convergente et divergente

Nous définissons ici les concepts de séquences multidimensionnelles convergentes et divergentes.

Définition 2.1 (Séquence divergente). Une g-k-séquence $\varsigma = \langle i_1, \dots i_g \rangle$ est divergente si $\forall e_{l_j} \in i_j \nexists e_{l'_{j'}} \in i_{j'}$, $i' < i$ tel que $e_{l_j} \subset e_{l'_{j'}}$.

En d'autres mots, pour tout item e_{k_j} de la séquence $\langle i_1, \dots, i_g \rangle$, il n'existe pas d'item plus général présent dans la séquence préfixe $\langle i_1, \dots, e_{j-1} \rangle$.

Enseigne	Date	items multidimensionnels
Enseigne 1	1	(Mntp,Pepsi)
	2	(Mntp,Pepsi),(Nice,Coca)
	3	(Mars.,Coca),(Mun.,Pepsi)
	4	(Moscou, Pepsi)
	5	(NY,Evian),(Pek.,Coca)
	6	(Ch.,Whisky)
Enseigne 2	1	(Mntp,Pepsi)
	2	(Nimes,Pepsi),(Mars.,Coca)
	3	(Dort.,Coca),(Nîmes,Pepsi)
	4	(Mun.,Coca)
	5	(Mntp,Evian),(LA,Coca)
	6	(Tok.,Whisky)
Enseigne 3	1	(NY,Bie.),(Ch,Whisky)
	2	(LA,Bie.)
	3	(SF,Bie.)
	4	(Pek.,Bie.),(Mntp,Vin)

TAB. 2.1 – Base de données exemple DB

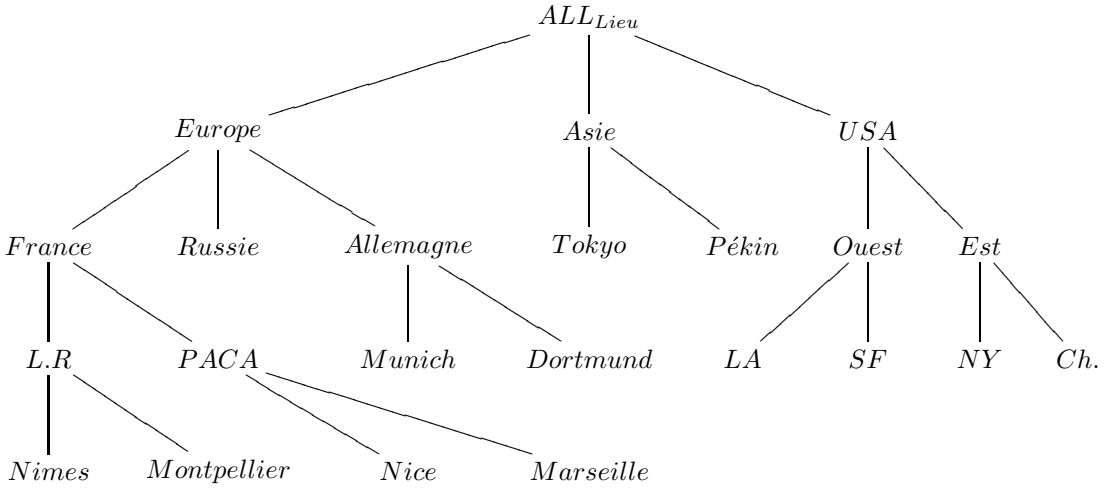


FIG. 2.1 – Hiérarchie sur Lieu

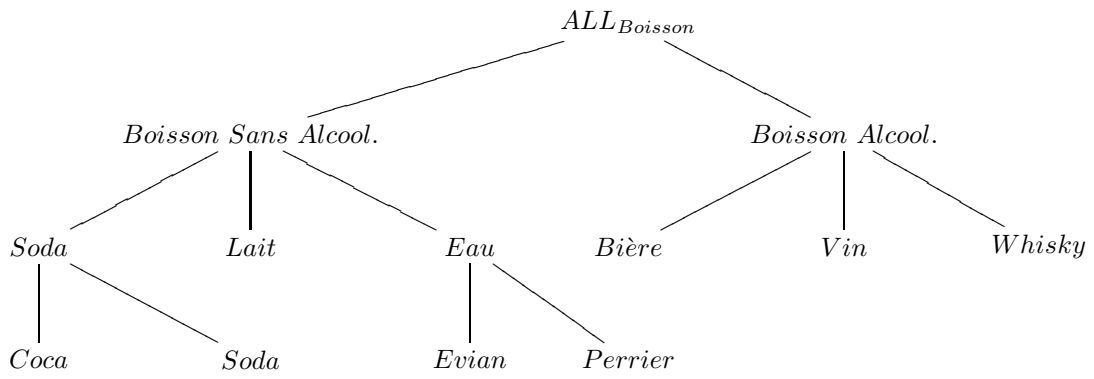


FIG. 2.2 – Hiérarchie sur les boissons

Par exemple, la séquence $\langle\{(Coca, Montpellier)\}, \{(Coca, LR)(Pepsi, PACA)\}, \{(Soda, France)$,-$(Soda, Allemagne)\}\rangle$ est une séquence multidimensionnelle divergente.

Définition 2.2 (Séquence convergente)**.** Une g-k-séquence $\varsigma = \langle i_1, \ldots i_g\rangle$ est convergente si $\forall e_{l_j} \in i_j \nexists e_{l'_{j'}} \in i_{j'}$, $i' < i$ tel que $e_{l'_{j'}} \subset e_{l_j}$.

Pour tout item e_{k_j} de la séquence $\langle i_1, \ldots, i_g\rangle$, il n'existe pas d'item plus spécifique présent dans la séquence préfixe $\langle i_1, \ldots, i_{j-1}\rangle$.

Ainsi, la séquence $\langle\{(Soda, Europe)(B.A, Asie)\}, \{(Coca, France)(Pepsi, Russie)\}\{(Coca, Mtp)\}\rangle$ est une séquence multidimensionnelle convergente.

Motifs séquentiels multidimensionnels convergents et divergents

A partir des définitions 2.1 et 2.2, nous pouvons définir le problème de l'extraction de motifs séquentiels multidimensionnels convergents ou divergents.

Définition 2.3. Etant donné un seuil de support minimum $minsupp$, le but de l'extraction de motifs séquentiels multidimensionnels convergents (respectivement divergents) dans un base de données DB dont l'ensemble des dimensions $\mathcal{D}$ est partitionné en quatre sous ensembles D_A, D_R, D_T et D_I est d'extraire l'ensemble complet des séquences multidimensionnelles convergentes (respectivement divergentes) dont le support est supérieur ou égal à $minsupp$.

Pour extraire de tels motifs, nous adoptons une approche de type « *pattern growth* ». Il suffit donc d'adapter l'algorithme 7 (CMSP_Cand p. 77). Ainsi, au lieu d'extraire de manière gloutonne toutes les séquences, nous nous focalisons à chaque pas sur les items les plus spécifiques. Ainsi seul l'algorithme d'extraction des items fréquents sur la base de données projetée est modifié. L'algorithme 14 permet

d'extraire seulement les items h-généralisés les plus spécifiques sur une base de données projetée par rapport à une séquence préfixe α.

La base projetée est parcourue une seule fois pour extraire tous les items fréquents les plus spécifiques. Deux types d'items peuvent être extraits :

1. Les items qui ne peuvent pas être inclus dans le dernier itemset de la séquence préfixe courante α. Ces items sont donc inclus dans un nouvel itemset de α. Pour extraire ces items et prendre en compte les items h-généralisés, nous devons étendre les transactions de la base projetée (pas à pas) avec la fonction LGS-CLosure.
2. Les items qui peuvent être inclus dans le dernier itemset de la séquence courante α. Dans ce cas, nous utilisons la fonction LGS-Closure$_X$ où X représente le dernier itemset de α.

Algorithme 14 : getFrequentItems

Données : Base projetée $DB|_\alpha$, $minsup$

Résultat : Ensemble F_l des items fréquents et maximalement spécifiques dans $DB|_\alpha$

début

```
/* Pour chaque séquence de données S_i de DB|_α nous avons :
   S_i = LGS-Closure_{lastItemset(α)}(same).otherTrans                    */
pour chaque S_i ∈ DB|α faire
    pour chaque item _e in same faire
        Gestion de _e ;
    pour chaque itemset is in other faire
        /* Recherche des items qui peuvent être insérés dans un nouvel
           itemset de α                                                   */
        SearchOtherTransFrequentItem e in LGS-Closure(is);
        /* Recherche des items qui peuvent être insérés dans le dernier
           itemset de α                                                   */
        si is supports lastItemset(α) alors
            SearchSameTransFrequentItem _e in LGS-Closure_{lastItemset(α)}(is);
retourner F_l = {e|support(e) ≥ suppmin ∧ e est maximalement spécifique};
```

fin

Ces différents algorithmes (CMSP_Cand p. 77 et l'algorithme 14) permettent l'extraction de motifs séquentiels multidimensionnels divergents. Pour extraire des motifs séquentiels multidimensionnels convergents, il est nécessaire d'utiliser les mêmes algorithmes mais sur une base de données inversée. En effet, il suffit d'inverser la relation d'ordre (commencer par la fin) au sein des séquences de données pour permettre l'extraction de connaissances allant du général au particulier.

Par exemple, afin d'extraire des motifs séquentiels multidimensionnels convergents, la séquence de données $\langle(1,1),(2,2),(3,3)\rangle$ sera inversée pour donner la séquence de données $\langle(3,3),(2,2),(1,1)\rangle$

sur laquelle des motifs séquentiels divergents seront extraits puis ensuite inversés afin d'être des motifs convergents.

2.1.1 Expérimentations

Dans cette section, nous reportons les expérimentations effectuées sur des jeux de données synthétiques et réels.

Données synthétiques

Les expérimentations ont été effectuées sur une base de données synthétiques composée de 100000 n-uplets définis sur 5 dimensions d'analyse. Des hiérarchies sont définies sur les dimensions d'analyse. Les expérimentations rapportent le nombre de fréquents obtenus et le temps d'exécution en fonction du support, du nombre de dimensions d'analyse, des spécificités des hiérarchies (degré et profondeur).

Les figures 2.3(c) et 2.3(d) montrent le nombre de séquences extraites et le temps d'exécution en fonction du degré des hiérarchies. Augmenter le degré d'une hiérarchie équivaut à spécialiser les données *horizontalement* (ajouter un élément à la fratrie). Notre approche permet de continuer à extraire des connaissances lorsque la hiérarchie se spécialise. Le temps de traitement devient cependant plus coûteux.

Les figures 2.3(a) et 2.3(b) montrent le nombre de fréquents extraits et le temps d'exécution en fonction de la profondeur des hiérarchies pour un seuil de support fixé. Etendre la hiérarchie d'un niveau engendre une spécialisation supplémentaire des données ($Soda$ devient $pepsi$ ou $coca$). Il y a ainsi plus de valeurs différentes dans la base de données. $M2S\_CD$ apporte une certaine robustesse face à ce phénomène de spécialisation. En effet, même si les données deviennent très détaillées (5 niveaux dans la hiérarchie), notre approche permet d'extraire des séquences définies sur plusieurs niveaux de hiérarchies. On remarque cependant que le temps de traitement est plus long quand le nombre de niveaux augmente. Ceci est dû au nombre d'items h-généralisés potentiellement fréquents qui augmente.

Les figures 2.4(a) et 2.4(b) montrent le temps d'exécution et le nombre de séquences extraites en fonction du support. Le nombre de fréquents augmente globalement lorsque le support diminue, ainsi que le coût de l'extraction. Néanmoins, il peut arriver que le nombre de fréquents diminue lorsque le support diminue. Ceci est dû au fait que des items plus spécifiques sont fréquents. Or un item plus général est plus rapidement fréquent dans une séquence de données. Il est alors possible d'obtenir moins de séquences.

Les figures 2.4(c) et 2.4(d) montrent le nombre de séquences extraites et le temps d'exécution en fonction du nombre de dimension d'analyse. Augmenter le nombre de dimensions d'analyse engendre une augmentation du nombre de fréquents et également du coût de l'extraction de tels motifs.

Ces expérimentations menées sur des données synthétiques montrent la robustesse de M2S_CD pour l'extraction des connaissances face à la diversité des données (nombre de dimensions, degré et profondeur des hiérarchies, etc.). Diversifier les données sources engendre un coût de traitement plus important qui reste cependant acceptable.

2.1.2 Données réelles

Nous avons étudié plusieurs parties du jeu *Eleusis* [Gar59]. Eleusis est un jeu de cartes dont le but est de trouver une règle secrète. Les règles secrètes sont des séquences de cartes contenant une partie droite et une partie gauche. Chaque partie peut contenir plusieurs cartes. Ce jeu permet de simuler la découverte scientifique qui est formée de tests, publications et réfutations. Nous avons donc analysé différentes parties du jeu développé[1] dans [DS05]. Nous avons décrit ce problème selon plusieurs dimensions d'analyse :

- une dimension organisant la valeur des cartes (figures, chiffres, impairs, pairs, etc.)
- une dimension organisant la couleur des cartes (rouge, noir, cœur, etc.)
- une dimension positionnant la carte dans la séquence (partie droite ou gauche)
- une dimension pour la réponse de l'oracle (exemple positif ou négatif).

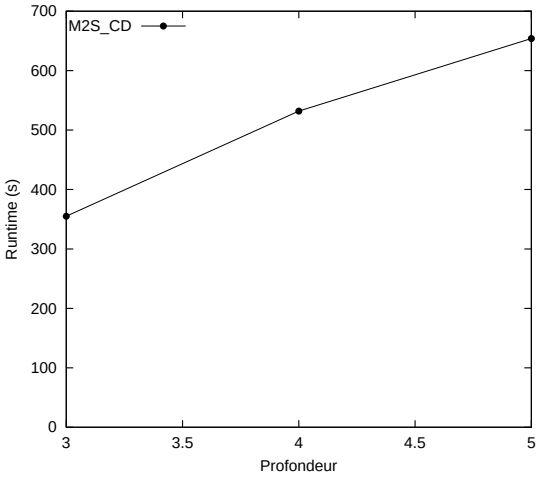
A l'aide de $M2S\_CD$, nous obtenons des séquences convergentes et divergentes sur ce jeu de données. Une des séquences divergentes extraites est : *Pour la règle nénuphar, les joueurs jouent fréquemment le trois de pique, puis l'as de pique avant de jouer un carte impaire de type pique puis une carte impaire de couleur noire*. Une des séquences convergentes extraites est : *Pour la règle lis, les joueurs proposent d'abord une carte rouge puis une carte de cœur et enfin une carte de type chiffre de couleur cœur*. Notons que ces règles, déclarées pertinentes par l'expert, n'auraient jamais pu être extraites à l'aide d'un algorithme classique.

2.2 Discussion

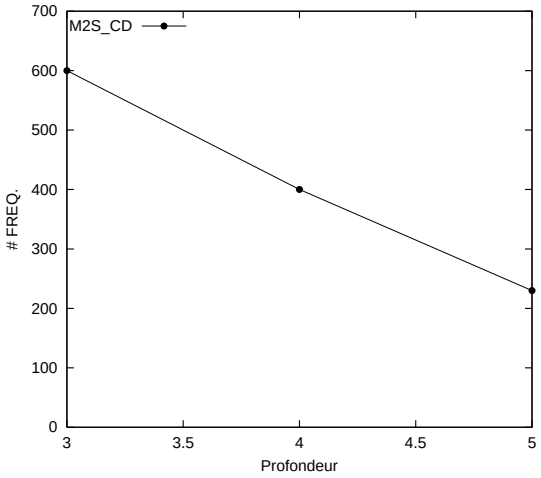
Dans ce chapitre, nous proposons une méthode originale pour extraire des connaissances multidimensionnelles définies sur plusieurs niveaux de hiérarchies mais selon un certain point de vue : du général au particulier ou vice et versa. Nous définissons ainsi le concept de séquences multidimensionnelles convergentes ou divergentes ainsi que les algorithmes associés basés sur le paradigme "pattern growth". Des expérimentations, sur des jeux de données synthétiques et réelles, montrent l'intérêt de notre approche $M2S\_CD$.

Il est nécessaire d'effectuer une analyse plus approfondie des connaissances extraites afin de ne retenir que les séquences qui sont réellement convergentes ou divergentes et d'éviter d'extraire des séquences qui sont *artificiellement* convergentes ou divergentes. Il serait également intéressant de tester notre proposition sur d'autres types de données réelles afin d'extraire de nouveaux types de connaissances comme la naissance d'une mode (études des achats), la découverte de pandémies (données hypocratiques) ou l'extraction d'article « fondateur » (données bibliographiques). A partir de ces nouveaux types de connaissances, nous pouvons extraire les top k séquences convergentes ou divergentes.

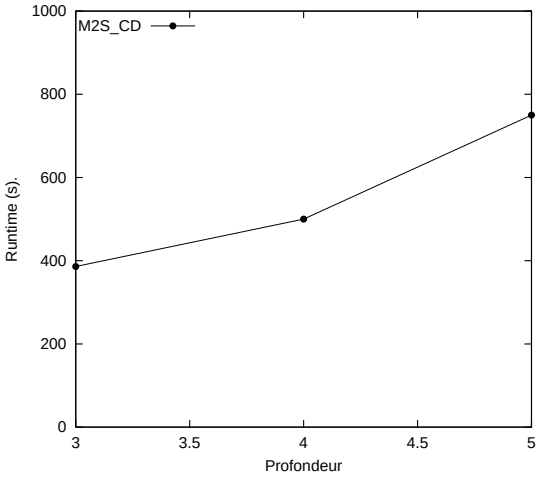
[1] http ://www.lirmm.fr/kayou/netoffice/eleusis/



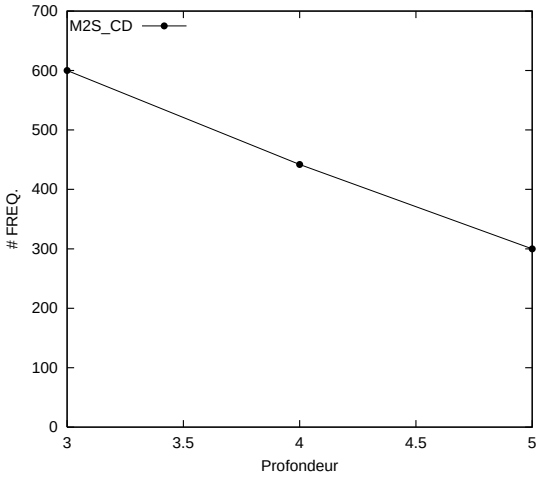
(a) Temps d'exécution en fonction de la profondeur des hiérarchies (Degré =3)



(b) #séquences extraites en fonction de la profondeur des hiérarchies (Degré =3)

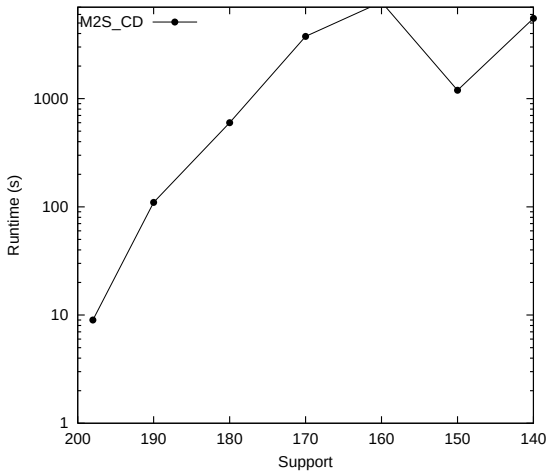


(c) Temps d'exécution en fonction du degré des hiérarchies (Profondeur =4)

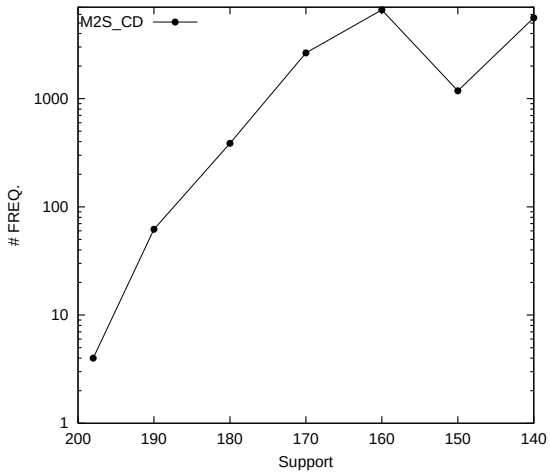


(d) #séquences extraites en fonction du degré des hiérarchies (Profondeur =4)

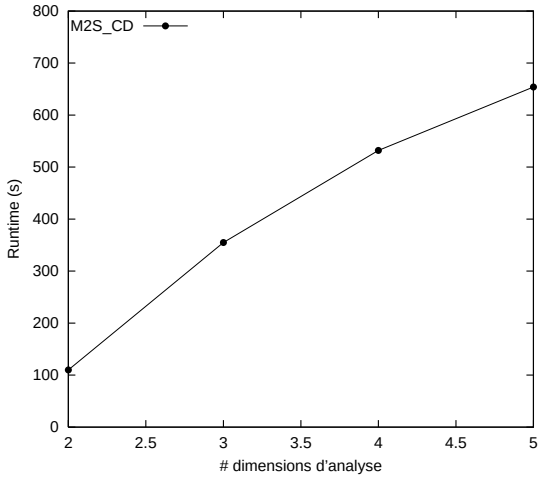
FIG. 2.3 – Expérimentations sur des données synthétiques en fonction des paramètres des hiérarchies



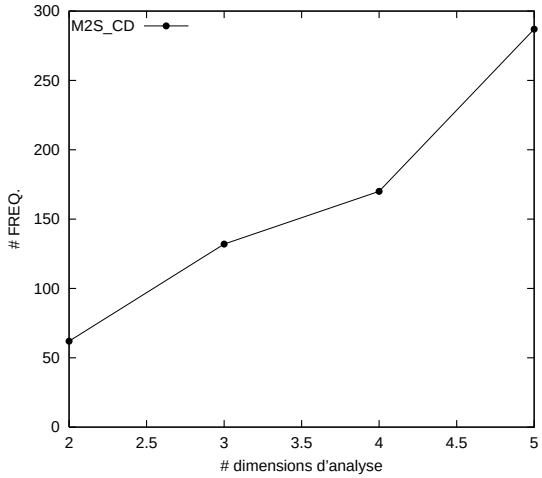
(a) Temps d'exécution en fonction du support (Profondeur =3, Degré =3)



(b) #séquences extraites en fonction du support (Profondeur =3, Degré =3)



(c) Temps d'exécution en fonction du nombre de dimensions d'analyse (Profondeur =3, Degré =3)



(d) #séquences extraites en fonction du nombre de dimensions d'analyse (Profondeur =3, Degré =3)

FIG. 2.4 – Expérimentations sur des données synthétiques en fonction de D_A et $minsupp$

Chapitre 3

Prise En Compte De La Mesure

3.1 Introduction

Dans les chapitres précédents, nous nous sommes intéressés à la prise en compte de certaines spécificités inhérentes aux bases de données multidimensionnelles comme la multidimensionnalité et la présence de hiérarchies. Or, les données multidimensionnelles possèdent une autre spécificité qu'il est nécessaire de prendre en compte : la *mesure*. La mesure est une dimension numérique représentant le résultat de l'agrégation des faits définis sur les autres dimensions. Cette dimension apporte des informations significatives qui peuvent permettre d'améliorer la qualité de connaissances extraites si elle est correctement prise en compte car la prise en compte d'une dimension numérique dans l'extraction de motifs est un problème en soit. En effet, les approches d'extraction de motifs (séquentiels ou non, multidimensionnels ou non) s'appuient bien souvent sur une gestion symbolique des données qu'elles analysent alors que celle-ci peuvent nécessiter un tout autre traitement.

Dans ce chapitre, nous proposons différentes façons de prendre en compte la mesure :

- Nous introduisons des contraintes sur la mesure afin de ne considérer que les cellules vérifiant cette condition.
- Nous proposons de discrétiser cette dimension particulière à l'aide de partitions strictes ou floues.
- Nous proposons également de prendre en compte la mesure pour calculer le support des séquences multidimensionnelles.

Ce chapitre est organisé de la façon suivante. Nous montrons d'abord les limites des algorithmes définis précédemment lorsqu'ils doivent considérer les mesures des cellules dans un cube de données. Nous présentons ensuite les travaux sur l'extraction de motifs qui essaient de prendre en compte des attributs numériques. Nous présentons ensuite nos trois propositions pour prendre en compte cette dimension numérique. Chaque proposition est validée par des expérimentations effectuées sur des jeux de données synthétiques ou réels. Enfin, dans la section 3.7, nous discutons nos différentes propositions et introduisons des perspectives associées à ces travaux.

3.2 Limites des motifs séquentiels multidimensionnels

Dans cette section, nous expliquons pourquoi les motifs séquentiels multidimensionnels (def. 1.10 p. 46) ne sont pas *directement* adaptables dès qu'une dimension particulière (*e.g.* la mesure) apparaît. Nous montrons par l'intermédiaire d'un exemple, qu'il est nécessaire de prendre en compte cette dimension.

Considérons une société de vente en ligne stockant les opérations de ses clients dans une base de données. Le tableau Tab. 3.1 représente plusieurs n-uplets de cette base de données. La partition de l'ensemble des dimensions $\mathcal{D}$ est la suivante : $D_I = \emptyset, D_R = \{\text{CID}\}, D_T = \{Date\}$ et $D_A = \{City, Cust\text{-}Grp, A\text{-}Grp, Product\}$

CID	Date	City	Customer Informations		Product
			Cust-Grp	Age-Grp	
C_1	1	NY	$Educ.$	$Middle$	A
C_1	1	NY	$Educ.$	$Middle$	B
C_1	2	LA	$Educ$	$Middle$	C
C_2	1	SF	$Prof.$	$Middle$	A
C_2	2	SF	$Prof.$	$Middle$	C
C_3	1	DC	$Business$	$Retired$	A
C_3	1	LA	$Business$	$Retired$	B

TAB. 3.1 – Table de faits

Les données de production des entreprises et des administrations, sont souvent agrégées dans un entrepôt de données à des fins d'analyse. Ainsi, une (ou plusieurs) dimension particulière appelée *mesure*, matérialise le résultat de cette agrégation. Cette dimension est numérique. Elle représente le résultat d'agrégation des données transactionnelles. La fonction d'agrégation dépend de la sémantique de l'application (somme, comptage, moyenne, etc.). Nous considérons, dans ce chapitre, un comptage comme opérateur d'agrégation.

Un cube de données peut donc être vu comme une application d'un ensemble de dimensions $\mathcal{D} = \{D_1, \ldots, D_n\}$ vers une dimension particulière M. Plus précisément, pour chaque n-uplet $(d_1, \ldots, d_n)$ défini sur $\mathcal{D}$, une valeur m de $Dom(M)$ est associée.

$$D_1 \times D_2 \times \ldots \times D_n \rightarrow M$$

$$(d_1, d_2, \ldots, d_n) \mapsto m$$

Le tableau Tab. 3.2 représente un exemple de cube de données résultant de l'agrégation de données transactionnelles issues de bases de données suivant le schéma de la table du tableau Tab. 3.1. Puisque les données sont agrégées dans une perspective d'analyse, la notion d'individu (CID) disparaît au profit de groupe d'individus (customer-grp, customer-age, etc.). De plus, une nouvelle dimension apparaît : la

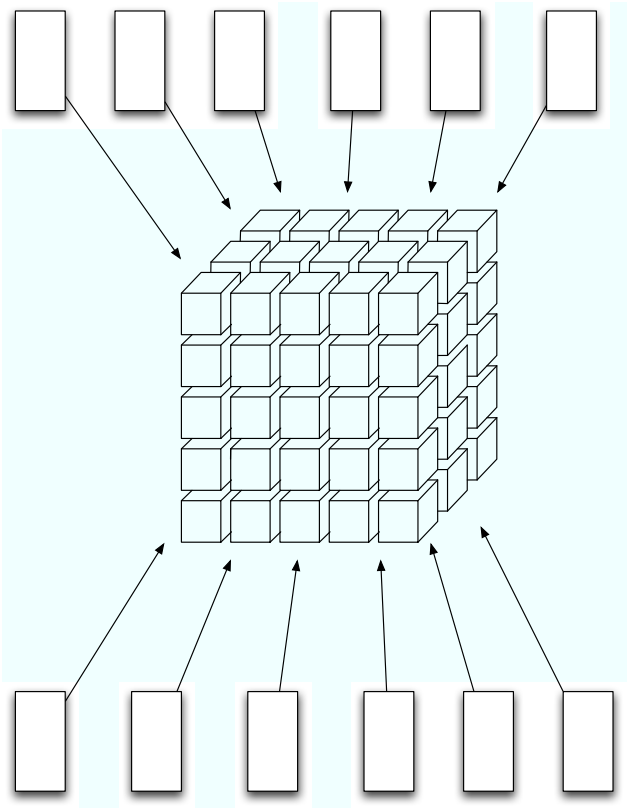


FIG. 3.1 – Agrégation des données de productions dans un cube de données

Date	City	Customer Informations		Product	Mesure
		Cust-Grp	Age-Grp		
1	NY	$Educ.$	$Middle$	A	123
1	NY	$Educ.$	$Middle$	B	234
2	LA	$Educ$	$Middle$	C	120
1	SF	$Prof.$	$Middle$	A	125
2	SF	$Prof.$	$Middle$	C	115
1	DC	$Business$	$Retired$	A	1
1	LA	$Business$	$Retired$	B	24

TAB. 3.2 – Cube de données « exemple »

mesure. Il est donc nécessaire de faire une nouvelle partition de l'ensemble des dimensions $\mathcal{D} \cup \{M\}$. Considérons la partition suivante :

- $D_T = \{Date\}$
- $D_R = \{Cust\text{-}Grp\}$
- $D_A = \{City, A\text{-}Grp, Product, Mesure\}$

Puisque la notion d'individu a disparu (CID), nous prenons comme dimension de référence le groupe de consommateurs. Cette dimension permet d'identifier 3 blocs comme l'illustre le tableau Tab. 3.3. En effet, D_R permet d'identifier les blocs $B_{educ.}, B_{prof.}$ et $B_{business}$. La relation d'ordre reste la même ($D_T = \{Date\}$).

La mesure est intégrée dans les dimensions d'analyse ($M \in D_A$). Par rapport aux définitions définies dans les chapitres précédents, il est assez intuitif de traiter cette dimension comme une dimension d'analyse et considérer seulement les cellules qui ont une mesure associée non vide.

Date	City	Customer Informations		Product	Mesure
		Cust-Grp	Age-Grp		
1	NY	**Educ.**	$Middle$	A	123
1	NY	**Educ.**	$Middle$	B	234
2	LA	**Educ.**	$Middle$	C	120
1	SF	**Prof.**	$Middle$	A	125
2	SF	**Prof.**	$Middle$	C	115
1	DC	**Business**	$Retired$	A	1
1	LA	**Business**	$Retired$	B	24

TAB. 3.3 – Partition en blocs en fonction de $D_R = \{Cust\text{-}Grp\}$

L'extraction de motifs séquentiels multidimensionnels s'appuie sur une gestion symbolique des données qu'elle traite. Ainsi, étant donnée la partition précédente, l'extraction de motifs séquentiels multidimensionnels a pour objectif de découvrir des corrélations entre la ville, l'âge des consommateurs, les produits vendus et la mesure associée au cours du temps. Cependant, les motifs extraits présentent des limites non négligeables dues à la gestion symbolique de la mesure. En effet, en se basant sur les définitions précédentes, nous pouvons obtenir les situations suivantes :

Support de la séquence $\langle\{(*, M, A, 125)\}\rangle$ Le support absolu de la séquence $\langle\{(*, M, A, 125)\}\rangle$ est égal à 1. En effet, seul le bloc $B_{Prof.}$ supporte cette séquence. Le bloc $B_{Educ.}$ contient une séquence relativement similaire $\langle\{(*, M, A, 123)\}\rangle$. Toutefois, la gestion symbolique de la mesure (dimension numérique) implique que les valeurs 123 et 125 sont considérées comme totalement différentes.

Support de la séquence $\langle\{(*, *, A, *)\}\rangle$ Le support absolu de la séquence $\langle\{(*, *, A, *)\}\rangle$ est égal à 3. Les trois blocs supportent donc la séquence. Plus précisément, les items des séquences de données

qui supportent la séquence (l'item) sont $(*, *, A, 123)$ pour $B_{Educ.}$, $(*, *, A, 125)$ pour $B_{Prof.}$ et $(*, *, A, 1)$ pour $B_{Business}$. Nous omettons les valeurs instanciées sur la ville, et l'âge afin de mettre en évidence l'observation suivante. $(*, *, A, 125)$ et $(*, *, A, 1)$ ont le *même impact* dans le calcul du support de la séquence $\langle\{(*, *, A, *)\}\rangle$.

Les deux points précédents soulignent les limites d'une gestion symbolique de la mesure dans l'extraction de motifs séquentiels multidimensionnels quand celle-ci est incluse dans les dimensions d'analyse. Il est donc nécessaire de prendre en compte la spécificité de cette dimension : son caractère numérique.

3.3 Panorama des travaux existants

La présence de valeurs numériques pour des « approches symboliques » est un problème relativement étudié.

Ainsi l'approche décrite dans [Lau03] propose une architecture basée sur les bases de données multidimensionnelles floues pour générer des résumés flous.

Dans [DHP03, DHP06], les auteurs s'intéressent à ce problème dans le cadre de l'extraction de règles d'association sur des attributs numériques.

[MRBM06] propose d'utiliser la mesure afin de calculer le support et la confiance des règles d'association multidimensionnelles dans des cubes de données. Ces règles identifient des corrélations entre les positions des cellules d'un cube de données. Elles ne permettent pas d'établir des corrélations entre différentes cellules.

Dans [FLT07], les auteurs utilisent la théorie des sous-ensembles flous pour prendre en compte les attributs numériques dans le contexte de la recherche de motifs séquentiels.

A notre connaissance, il n'existe pas d'approche qui s'attaque à la prise en compte de la mesure et son caractère numérique dans l'extraction de motifs séquentiels multidimensionnels. Dans ce chapitre, nous proposons trois façons de prendre en compte cette dimension :

1. En introduisant des *contraintes d'agrégats* sur les valeurs de mesure des cellules du cube, ce qui permet de réduire l'espace de recherche et de ne pas considérer les cellules dont la mesure associée ne respecte pas la contrainte.
2. En *discrétisant* la mesure à l'aide de partitionnements stricts ou flous afin de considérer la mesure comme une dimension d'analyse. Ceci permet de réduire la taille du domaine de la mesure, et également de considérer des valeurs similaires comme identiques.
3. En utilisant directement la puissance agrégative de la cellule pour calculer le *support* des séquences multidimensionnelles, ce qui nous amène à définir deux nouvelles méthodes pour calculer le support relatif d'une séquence multidimensionnelles.

Les sections suivantes traitent de chacune de ces propositions.

3.4 Contraintes d'agrégats sur la mesure

Afin de ne pas considérer les cellules du cube de données dont la mesure associée est trop faible, nous pouvons poser des *contraintes d'agrégats* sur la valeur de mesure.

Une contrainte d'agrégat évalue la qualité d'une cellule ou d'un motif au regard d'une mesure d'intérêt. La contrainte d'agrégat la plus utilisée est certainement la contrainte de *support minimum* pour l'extraction de motifs. Ici, la contrainte d'agrégat n'a pas pour objectif premier d'améliorer la qualité des motifs extraits. Nous utilisons ici la contrainte d'agrégat sur la mesure pour réduire l'espace de recherche et éviter que des cellules dont la mesure associée est trop faible supporte une séquence multidimensionnelle.

La forme caractéristique d'une contrainte d'agrégat est $m(X)\theta seuil$ où m est une fonction d'agrégat et $\theta \in \{<, \leq, =, \geq, >\}$. Par rapport à notre contexte, $m(X)$ sera la mesure m d'une cellule X du cube de données, l'opérateur θ le plus utilisé sera $\geq$. Toutefois, il est tout à fait possible de se focaliser uniquement sur les cellules dont la mesure associée est inférieure à un seuil considéré dans la perspective d'extraire des motifs séquentiels multidimensionnels sur les cellules faibles afin d'exhiber des modèles généraux sur de telles cellules et de comparer ces modèles à ceux obtenus à partir des cellules dont les mesures associées sont plus importantes.

Date	City	Customer Informations		Product	Mesure
1	NY	**Educ.**	$Middle$	A	123
1	NY	**Educ.**	$Middle$	B	234
2	LA	**Educ.**	$Middle$	C	120
1	SF	**Prof.**	$Middle$	A	125
2	SF	**Prof.**	$Middle$	C	115
1	DC	**Business**	$Retired$	A	**1**
1	LA	**Business**	$Retired$	B	**24**

TAB. 3.4 – Contraintes d'agrégats sur la mesure ($m \geq 50$)

Exemple 3.1. Par rapport au cube de données exemple (Tab. 3.3), on considère la contrainte d'agrégat $m \geq 50$. Les cellules du bloc $B_{Business}$ ne vérifient pas cette contrainte (Tab. 3.4. Le support relatif de la séquence $s = \langle\{(*, *, A, *)\}\rangle$ est $\frac{2}{3}$ puisque le bloc $B_{Business}$ ne supporte plus la séquence s avec la prise en compte de la contrainte $m \geq 50$. En effet, la séquence de données associée au bloc $B_{Business}$ est la séquence vide $\langle\rangle$ car aucune cellule du bloc $B_{Business}$ ne vérifie la contrainte $m \geq 50$.

La prise en compte d'une contrainte d'agrégat sur la mesure peut être effectuée soit dans un prétraitement des cellules du cube de données, soit au cours de l'extraction des motifs séquentiels multidimensionnels en ajoutant cette condition dans les algorithmes d'extraction.

Expérimentations

Nous avons mené des expérimentations sur des données réelles (cube de données d'EDF). Ces expérimentations visent à mesurer la taille de l'espace de recherche par rapport à la contrainte d'agrégat utilisée. Nous examinons ainsi le nombre d'items fréquents les plus spécifiques extraits par l'algorithme M^2SP en fonction de la condition d'iceberg considérée. La figure 3.2 décrit cette expérimentation.

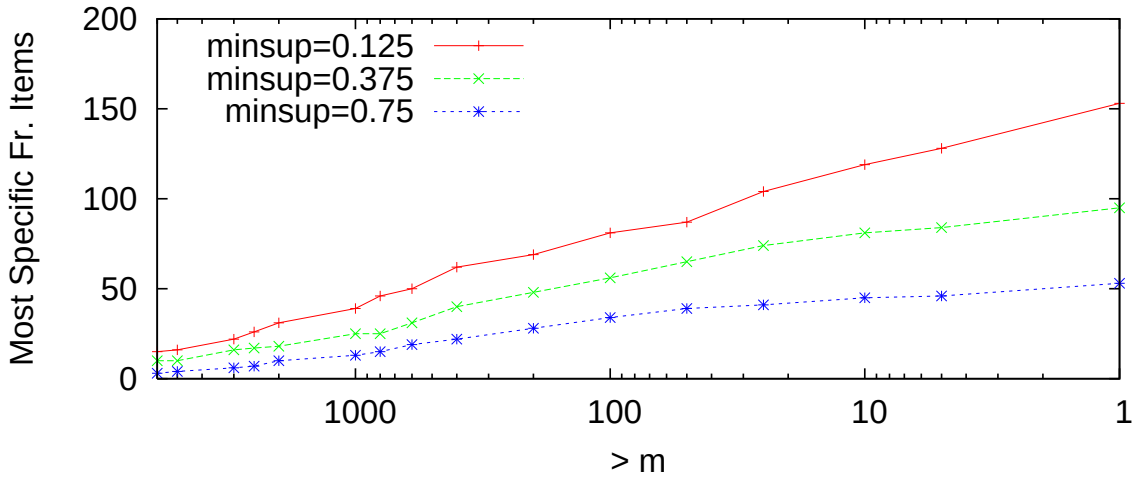


FIG. 3.2 – Nombre d'items fréquents les plus spécifiques en fonction de la contrainte d'agrégat

Plus la condition d'iceberg utilisée est contraignante, moins le nombre d'items fréquents les plus spécifiques est important. Ceci vient du fait, que la contrainte d'agrégats permet de ne pas considérer les cellules dont la mesure associée est inférieure à la valeur m. Ainsi, les positions présentes sur ces cellules ne sont pas considérées. La taille de l'espace de recherche est réduite. Cet observation se manifeste par le fait que le nombre des items multidimensionnels fréquents les plus spécifiques varie sensiblement en fonction de la contrainte d'agrégat. Cette contrainte évite ainsi de considérer les cellules dont la mesure est trop faible, ce qui a un impact dans le support des items multidimensionnels.

L'introduction d'une contrainte d'agrégat sur la mesure permet de ne pas considérer toutes les cellules du cube de données mais seulement celles qui vérifient la condition. Par exemple, une telle condition évite de considérer les cellules du cube de données dont la mesure associée est jugée trop faible. Même si l'introduction d'une contrainte d'agrégat sur la mesure permet de réduire l'espace de rechercher et d'obtenir également des motifs plus intéressants, cette approche ne permet pas d'exploiter pleinement *la puissance informationnelle* de la valeur de mesure. Par exemple, il n'est pas possible, avec cette seule proposition, d'extraire des connaissances où un item est défini avec une mesure forte et un autre item avec des mesures plus faibles. La section suivante s'attaque à la prise en compte de ces informations à l'aide d'un partitionnement de la mesure.

3.5 Discrétisation du domaine de la mesure

Dans cette section, nous proposons de discrétiser la mesure afin de bénéficier des informations présentes sur cette dimension. Lors de l'extraction de motifs séquentiels multidimensionnels, cette dimension peut alors être considérée de la même façon que les autres. La discrétisation d'un domaine de valeurs numériques peut se faire de plusieurs façons. Nous étudions ici différentes partitions possibles et comparons les motifs séquentiels multidimensionnels extraits selon la discrétisation opérée (partition de la mesure en intervalles stricts ou en sous-ensembles flous) et le comptage utilisé (normal ou flou).

3.5.1 Partition en intervalles stricts

Dans le cadre de l'extraction de connaissances par des techniques *symboliques* sur des données numériques, plusieurs approches ont été proposées afin de discrétiser les domaines de définition des attributs numériques en intervalles distincts. Il s'agit, la plupart du temps, de définir les bornes des intervalles de façon automatique. Plusieurs types de partitions sont couramment utilisés :

- Découpage *equi-width* où les intervalles ont tous la même largeur.
- Découpage *equi-depth* qui assure une équi-répartition des enregistrements dans chaque intervalle.
- Découpage selon la connaissance d'un expert ou le résultats de calculs statistiques.

La plupart des propositions qui s'attaquent à la découverte de motifs dans des données numériques à l'aide d'une partition en intervalles stricts [KLNS04, SA96b] soulignent la difficulté de déterminer les bornes optimales et le nombre d'intervalles. Des intervalles mal définis ont des conséquences sur la qualité des données extraites.

Par rapport au cube de données exemple 3.3, nous choisissons la partition du domaine de la mesure en trois intervalles distincts :

- $Peu = [0, 99]$
- $Moyen = [100, 199]$
- $Beaucoup = [200, 300]$

Ainsi, chaque valeur m de mesure d'une cellule est associée à un unique intervalle parmi les trois définis. Le tableau Tab. 3.5 illustre le cube de données exemple après discrétisation de la mesure.

Le support absolu de la séquence $\langle\{(*, Middle, A, Moyen)\}\rangle$ est égal à 2. Avec la discrétisation des valeurs de mesure, les valeurs 123 et 125 appartiennent au même intervalle ($Moyen$) et sont donc considérer comme similaire lors de l'extraction de motifs séquentiels multidimensionnels. C'est ainsi que le bloc $B_{Educ.}$ supporte désormais la séquence $\langle\{(*, Middle, A, Moyen)\}\rangle$.

Expérimentations

Pour montrer l'utilité d'un partitionnement strict, nous avons mené des expérimentations sur des données réelles (cube EDF). Nous étudions le nombre des items fréquents les plus spécifiques. Afin

Date	City	Customer Informations		Product	Mesure		
					Peu	$Moyen$	$Beaucoup$
1	NY	**Educ.**	$Middle$	A	0	1	0
1	NY	**Educ.**	$Middle$	B	0	0	1
2	LA	**Educ.**	$Middle$	C	0	1	0
1	SF	**Prof.**	$Middle$	A	0	1	0
2	SF	**Prof.**	$Middle$	C	0	1	0
1	DC	**Business**	$Retired$	A	1	0	0
1	LA	**Business**	$Retired$	B	1	0	0

TAB. 3.5 – Partitions strictes des valeurs de la mesure

d'exploiter pleinement la « puissance informationnelle » de la mesure, nous interdisons la valeur joker sur cette dimension. La figure 3.3 montre la distribution des valeurs de mesures des cellules du cube de données. Le domaine de définition est large (de 1 à environ 47000). Les mesures inférieures à 10 sont celles qui sont le plus souvent associées aux cellules du cube de données.

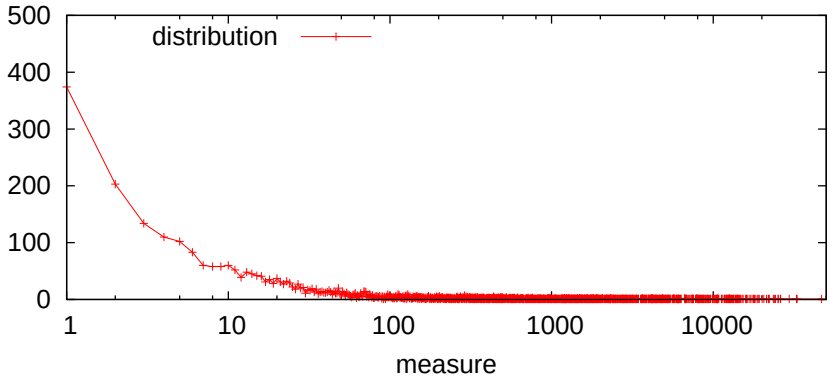
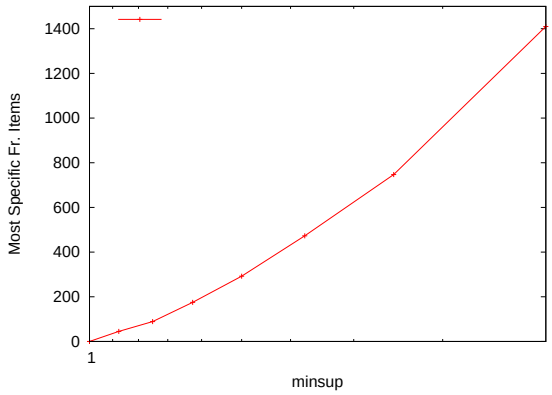


FIG. 3.3 – Distribution des données en fonction de la mesure

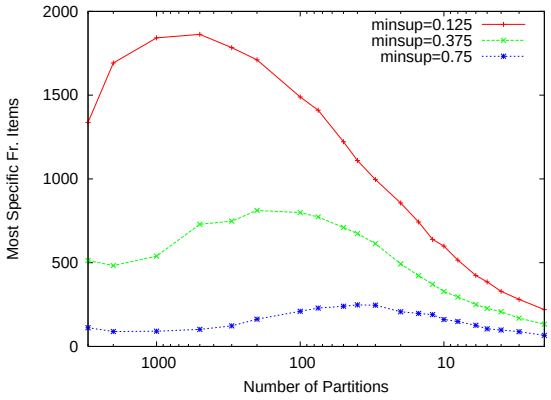
La figure 3.4(a) rapporte le nombre d'items fréquents les plus spécifiques en fonction du seuil de support considéré tout en interdisant la valeur joker sur la mesure. Cette courbe souligne la difficulté d'extraire des connaissances sans aucun traitement de la mesure. En effet, si le seuil de support est trop fort, le nombre d'items fréquents dont la dimension mesure n'est pas intanciée par * est très faible, voire nul. Par contre, lorsque le support devient faible, le nombre d'items devient relativement important. Il est donc nécessaire de partitionner la mesure. Nous utilisons dans ces expérimentations des partitions *equi-depth*, c'est à dire que chaque partition possède le même nombre d'éléments.

La figure 3.4(b) rapporte le nombre d'items fréquents les plus spécifiques (* interdite sur la mesure) en fonction du nombre de partitions considérées. Lorsque le nombre de partitions diminue (entre 3000 et 300), le nombre d'items fréquents les plus spécifiques augmente. Lorsque le nombre de partitions est très

élevé, on se retrouve dans le même cas que lorsque la mesure n'est pas partitionnée (difficile de trouver des items fréquents avec valeur joker interdite sur la mesure). Quand le nombre de partition diminue, des items (a, p) et (a, p') ($a \in D_A \setminus \{M\}$, p et p' sont des partitions de la mesure) apparaissent. En effet, à une combinaison a fréquente sur $D_A \setminus \{M\}$, on peut associer plusieurs éléments p issus la partition de la mesure. Le nombre d'items fréquents les plus spécifiques diminue ensuite lorsque le nombre de partition diminue (entre 300 et 2) car p et p' sont regroupés dans la même partition.



(a) Nombre d'items fréquents les plus spécifiques en fonction du seuil de support minimum (* interdite sur la mesure)



(b) Nombre d'items fréquents les plus spécifiques

FIG. 3.4 – Partitionnement Strict

Ces expérimentations montrent l'intérêt d'utiliser un partitionnement de la mesure. Toutefois, un tel partitionnement peut également créer des *effets de bords*. Ces effets de bords sont dus à un découpage strict des intervalles. Ainsi, si on considère X comme étant la limite entre deux intervalles, deux valeurs similaires $X - 1$ et $X + 1$ seront considérées comme différentes car elles appartiennent à deux intervalles différents. Une *partition floue* de la mesure permet d'atténuer ce problème et de découvrir des *motifs absents* avec un découpage strict de la mesure. Une partition floue permet également l'utilisation de plusieurs techniques de comptage.

3.5.2 Partition en sous-ensembles flous

Les ensembles flous ont été introduits afin de modéliser la représentation humaine des connaissances, et ainsi améliorer les performances des systèmes de décision qui utilisent cette modélisation.

Une sous-ensemble flou A de B est caractérisé par une application de B dans $[0, 1]$. Cette application, appelée *fonction d'appartenance* et notée μ_A représente le degré de validité de la proposition « x *appartient à* A » pour chacun des éléments x de B. Si $\mu_A(x) = 1$, l'objet x appartient totalement à A, et si $\mu_A(x) = 0$, il ne lui appartient pas du tout. Pour un élément x donné, la valeur de la fonction d'appartenance $\mu_A(x)$ est appelée degré d'appartenance de l'élément x au sous-ensemble A.

Par exemple, la figure 3.5 représente l'appartenance des éléments x de âge au sous-ensemble flou $Jeune$.

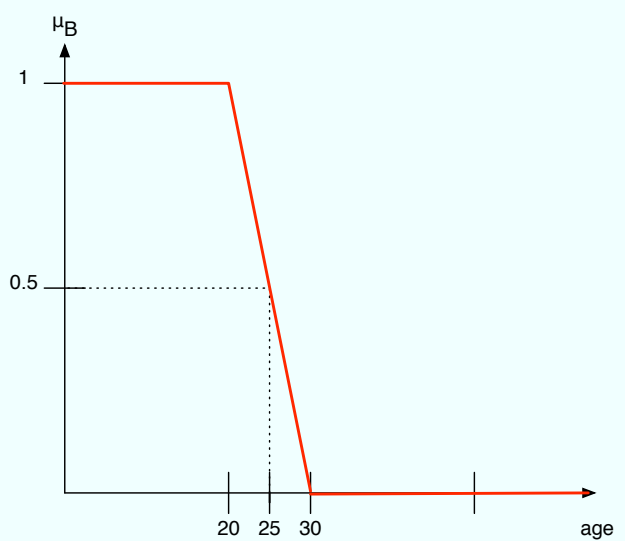


FIG. 3.5 – Exemple : sous-ensemble flou *jeune* sur l'attribut *âge*

De nombreuses propositions ont été formulées afin d'utiliser des partitions floues d'attributs numériques en vue d'une extraction de connaissances symboliques [FLT07, CA97]. Ces travaux utilisent différentes techniques afin de réaliser le partitionnement flou (savoir d'un expert, equi-depth, equi-width, clustering [HK01]).

L'utilisation d'une partition floue permet d'utiliser différentes méthodes de calcul du support d'une séquence comme développé dans [FLT07]. On peut ainsi *pondérer* la présence d'un item multidimensionnel par le degré d'appartenance de la mesure à la valeur symbolique considérée. Le support d'une séquence multidimensionnelle correspond à la moyenne pour tous les blocs de B_{DB,D_R} du degré d'appartenance de la séquence à chaque bloc. Ce degré est calculé en considérant l'intersection des sous-ensembles flous pour chaque item chaque itemset (utilisation d'une t-norme). Pour chaque bloc, la meilleure représentation sera renvoyé (utilisation d'une t-conorme).

Par rapport au cube de données exemple, la figure 3.6 illustre la partition floue en trois sous-ensembles *Peu, Moyen et Beaucoup* du domaine de la mesure. A partir de ces sous-ensembles flous, l'extraction des motifs séquentiels multidimensionnels s'effectue maintenant sur la base illustrée Tab. 3.6.

3.6 La mesure pour calculer le support

Dans la plupart des cas, les valeurs des agrégats d'un cube de données peuvent être vues comme un pré-calcul du support de certaines séquences. En effet, une cellule peut être vue comme une séquence d'un item (défini sur les dimensions d'analyse de la cellule). La mesure de la cellule quantifie l'aptitude de la cellule à supporter l'item. Ainsi, une cellule dont la mesure associée est 100 ne doit pas être considérée

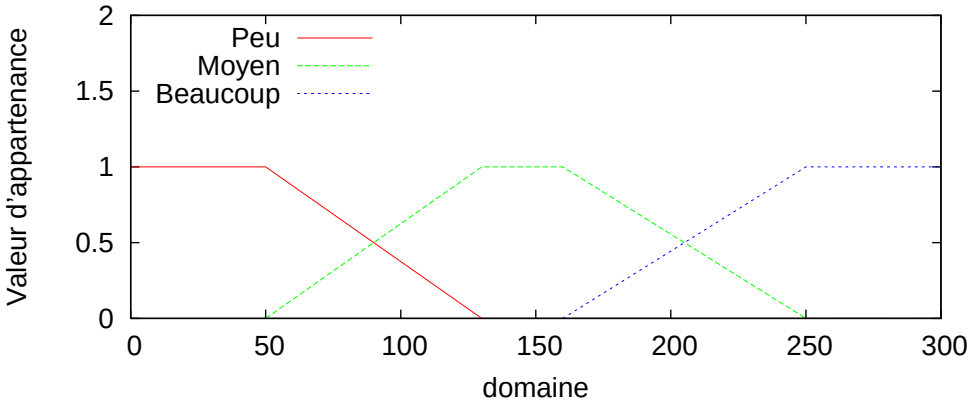


FIG. 3.6 – Partitionnement flou de la mesure

Date	City	Customer Informations		Product	Mesure		
					Peu	$Moyen$	$Beaucoup$
1	NY	**Educ.**	$Middle$	A	0.0875	0.925	0
1	NY	**Educ.**	$Middle$	B	0	0.18	0.82
2	LA	**Educ.**	$Middle$	C	0.125	0.875	0
1	SF	**Prof.**	$Middle$	A	0.0625	0.9375	0
2	SF	**Prof.**	$Middle$	C	0.1875	0.8125	0
1	DC	**Business**	$Retired$	A	1	0	0
1	LA	**Business**	$Retired$	B	1	0	0

TAB. 3.6 – Sous-ensembles flous sur les valeurs de la mesure

de la même façon qu'une cellule qui a une mesure nettement inférieure. Nous proposons ici de prendre en compte la valeur des agrégats afin de calculer le support des séquences multidimensionnelles.

Il est nécessaire de maintenir l'ordre d'apparition des événements dans la séquence ainsi que l'une des propriétés fondamentales inhérentes à l'extraction de motifs[1] (multidimensionnels ou non) : *l'antimonotonie du support*. Soit un motif p, quel que soit P, un super motif de p, on a :

$$support(p) \geq support(P)$$

Tous les algorithmes d'extraction de motifs se basent sur cette propriété afin de parcourir efficacement l'espace de recherche pour extraire tous les motifs fréquents. Ainsi, ils partent de la séquence vide $\langle\rangle$ et essaient d'extraire des séquences plus longues, soit par un parcours niveau par niveau (*a priori*, [AS95, MCP98]), soit en profondeur d'abord (classes d'équivalence [Zak01], Pattern-growth [PHMA$^+$04]). L'extraction de motifs séquentiels (multidimensionnels) a pour objectif d'établir des corrélations entre des événements suivant leur chronologie d'apparition. Il est ainsi nécessaire de maintenir l'ordre en les éléments d'une séquences.

Pour préserver l'ordre d'apparition des événements dans la séquence ainsi que l'antimonotonie du support, nous utilisons une *t-norme* $\otimes$ qui est une généralisation de la conjonction logique. Une t-norme est un opérateur $[0,1] \times [0,1] \rightarrow [0,1]$ qui est associatif, commutatif, monotone et qui satisfait les conditions s $\alpha \otimes 0 = 0$ et $\alpha \otimes 1 = \alpha$. Les exemples les plus connus de t-norme sont le minimum $(\alpha, \beta) \mapsto \min(\alpha, \beta)$, le produit $(\alpha, \beta) \mapsto \alpha\beta$ et la t-norme de Lukasiewicz $(\alpha, \beta) \mapsto \max(\alpha + \beta - 1, 0)$. Nous utiliserons ici le $\min$ comme t-norme dans les exemples.

Nous utilisons également une *t-conorme* $\oplus$ qui correspond à une disjonction logique. $\oplus$ est un opérateur $[0,1]\times[0,1] \rightarrow [0,1]$ qui est associatif, commutatif, monotone et qui satisfait les conditions $\alpha \oplus 1 = 1$ et $\alpha \oplus 0 = \alpha$. Les exemples les plus connus de t-conorme sont le maximum $(\alpha, \beta) \mapsto \max(\alpha, \beta)$, la somme probabiliste $(\alpha, \beta) \mapsto \alpha + \beta - \alpha\beta$, la somme bornée $(\alpha, \beta) \mapsto \min(\alpha + \beta, 1)$, etc. Nous utiliserons ici le $\max$ comme t-conorme dans nos exemples.

Etant donné qu'une séquence peut apparaître plusieurs fois dans une séquence de données identifiée par un bloc, il est nécessaire d'exhiber la combinaison qui « supporte le mieux » la séquence. Plus précisément, il faut exhiber les cellules qui ont la plus forte valeur de mesure et qui permettent de supporter la séquence.

Pour calculer le support relatif d'une séquence multidimensionnelle, nous avons deux possibilités :

1. L'utilisateur peut considérer que l'importance des blocs doit s'exprimer dans le calcul du support d'une séquence. Ainsi, les blocs ont des poids différents en fonction de leur **effectif** ou **population**. L'importance d'un bloc intervient dans la valeur du support de la séquence. Par exemple, un bloc important à un impact plus important dans le support d'une séquence qu'un bloc de poids faible.

[1] Dans le contexte des motifs séquentiels multidimensionnels, un super motif sera un motif plus *spécifique* ($\prec_S$, voir définition 3.2 page 69).

2. Comme pour les motifs séquentiels classiques où les dimensions de références D_R sont réduites à un singleton représentant l'identifiant d'une séquence de données (*e.g.* l'identifiant du client dans le contexte de l'analyse du panier de la ménagère), les blocs peuvent avoir des impacts égaux dans le support d'une séquence, et ceci quelque soit leur effectif.

Nous définissons ainsi deux façons de calculer le support relatif d'une séquence dans un cube de données suivant les deux points décrits précédemment.

Micro count prend en compte l'importance de chaque bloc lors du calcul du support d'une séquence. Ainsi, la mesure des cellules d'un bloc qui participent à supporter la séquence ($m[B_r, s_{i_j}]$) est divisée par la mesure totale ($m[cell(*, *, \ldots, *)]$).

Définition 3.1 (Micro Count).
Soit une g-k-séquence $s = \langle s_1, s_2, \ldots, s_g \rangle$, le support relatif de s dans un cube de données DB avec la technique micro count est égal à :

$$\text{Relative support}(s) = \sum_{\mathcal{B}_r \in B_{DB,D_R}} \bigoplus \bigotimes_{s_i \in s} \bigotimes_{s_{i_j} \in s_i} \frac{(m[B_r, s_{i_j}])}{m[(*, *, \ldots, *)]}$$

Pour chaque séquence apparaissant dans un bloc (tous les items de tous les itemsets doivent être présents en respectant la relation d'ordre), nous prenons la valeur minimale (t-norme) de la valeur de la mesure parmi les cellules supportant les items de la séquence (permet de garantir l'antimonotonie du support).

Puisqu'une séquence peut apparaître plusieurs fois dans la séquence de données pointée par la bloc, il faut considérer la meilleure solution, c'est-à-dire la combinaison la plus prometteuse. C'est pour cela que le support maximum (t-conorme) de cette séquence dans le bloc est retenu.

Macro count vise à calculer le support relatif d'une séquence en considérant que chaque bloc du cube de données doit avoir le même impact dans le support d'une séquence. Ainsi, la mesure des cellules d'un bloc B_r permettant à B_r de supporter la séquence recherchée ($m[B_r, s_{i_j}]$) est divisée par la valeur de mesure associée à B_r ($m[r, *, *, \ldots, *]$).

Définition 3.2 (Macro Count).
Soit une g-k-séquence $s = \langle s_1, s_2, \ldots, s_g \rangle$, le support relatif de s dans un cube de données DB avec la technique macro count est égale à :

$$\text{Relative support}(s) = \frac{1}{|B_{DB,D_R}|} \times \sum_{\mathcal{B}_r \in B_{DB,D_R}} \bigoplus \bigotimes_{s_i \in s} \bigotimes_{s_{i_j} \in s_i} \frac{(m[B_r, s_{i_j}])}{m[(r, *, \ldots, *)]}$$

Comme pour la définition 3.1, il faut rechercher la meilleure combinaison de cellules (t-conorme) afin que le support de la séquence dans le bloc soit maximal. Pour chaque combinaison, il faut garantir

l'antimonotonie du support, la valeur de mesure la plus faible (t-norme) des cellules de la combinaison est retenue.

Le calcul du support des items contenant une ou plusieurs valeurs jokers est assez simple. En effet, puisque nous considérons les mesures associées des cellules qui contiennent au plus un item de la séquence pour un bloc donné afin de calculer le support de la séquence. Ainsi, lorsqu'une valeur joker est présente dans un item de le séquence, il faut récupérer la mesure maximale parmi les cellules qui supportent cet item (exhiber la date où la mesure est maximale).

L'exemple 3.2 illustre le calcul des supports relatifs de plusieurs séquences en fonction de micro count et de macro count. La mesure considérée dans cet exemple est un dénombrement ($count$) sur des dimensions additives.

Exemple 3.2. Soit le cube de données présenté dans la figure 3.2. La partition des dimensions est la suivante :

- $D_T = \{Date\}$
- $D_R = \{Cust\text{-}Grp\}$
- $D_A = \{City, A\text{-}Grp, Product\}$

Le tableau Tab. 3.7 représente la mesure associée aux différent blocs B_r de B_{DB,D_R} ainsi que la mesure totale.

	D_R	D_A			M
$m[(Educ, *, *, *)]$	Educ.	*	*	*	477
$m[(Prof, *, *, *)]$	Prof.	*	*	*	240
$m[(Business, *, *, *)]$	Business	*	*	*	25
$m[(*, *, *, *)]$	*.	*	*	*	742

TAB. 3.7 – Les valeurs des différents blocs et la mesure totale

B_{Educ}				
1	NY	$Middle$	A	123
1	NY	$Middle$	B	234
2	LA	$Middle$	C	120

B_{Prof}				
1	SF	$Middle$	A	125
2	SF	$Middle$	C	115

$B_{Business}$				
1	DC	$Retired$	A	1
1	LA	$Retired$	B	24

TAB. 3.8 – DB partitionnée en 3 blocs par rapport à $D_R = \{Cust\text{-}Grp\}$

La séquence $\langle\{(*, *, A)\}\rangle$ est présente dans les 3 différents blocs. Le bloc B_{Educ} la supporte par l'intermédiaire de la cellule $\langle(1, NY, Middle, A) : 123\rangle$, les blocs $B_{Prof.}$ et $B_{Business}$ supportent également la séquence avec respectivement les cellules $\langle(1, SF, Middle, A) : 125\rangle$ et $\langle(1, LA, Retired, A) : 1\rangle$. Dans chaque bloc, cette séquence n'apparaît qu'une seule fois, la t-conorme sera ainsi appliquée sur un seul élément.

Pour MicroCount, on divise la mesure des cellules par la mesure totale : $\frac{125+123+1}{742} = 0,34$.

Séquences	MicroCount	MacroCount
$\langle\{(*,*,A)\}\rangle$	0.34	$0,27$
$\langle\{(*,Middle,*)\}\rangle$	$0,65$	$0,42$
$\langle\{(*,*,A)\},\{(*,Middle,*)\}\rangle$	0.32	$0,24$
$\langle\{(*,*,A),(*,*,B)\}\rangle$	$0,17$	$0,10$

TAB. 3.9 – Le support relatif de plusieurs séquences pour MicroCount et MacroCount

Pour MacroCount, on divise la mesure de chaque cellule par la mesure associée au bloc contenant la cellule, puis on calcule la moyenne : $\frac{\frac{123}{477}+\frac{125}{240}+\frac{1}{25}}{3} = 0,27$.

Nous remarquons que pour le comptage avec MicroCount, les blocs $B_{Educ.}$, $B_{Prof.}$ et $B_{Business}$ ont des influences respectives d'environ 64%, 32% et 4% dans le calcul du support d'une séquence alors qu'avec le comptage MacroCount, ils ont tous des influences égales.

La séquence $\langle\{(*,Middle,*)\}\rangle$ est uniquement présente dans les blocs $B_{Educ.}$ et $B_{Prof.}$. Elle apparaît plusieurs fois dans ces deux blocs. En effet, cette séquence est supportée aux dates 1 et 2 pour les deux blocs. La t-conorme va nous permettre de retenir la meilleure solution pour chaque bloc. Pour le MicroCount, le support de la séquence $\langle\{(*,Middle,*)\}\rangle$ est égal à : $\frac{\max(357,120)+\max(125,115)}{742} = 0,65$.

La séquence $\langle\{(*,*,A)\},\{(*,Middle,*)\}\rangle$ est également présente dans les blocs $B_{Educ.}$ et $B_{Prof.}$. Une seule combinaison est possible pour chacun de ces deux blocs.

Par exemple, les cellules $\langle(1,NY,Middle,A) : 123\rangle$ et $\langle(1,LA,Middle,C) : 120\rangle$ permettent à $B_{Educ.}$ de supporter la séquence. La t-norme (min) va nous permettre de garantir l'antimonotonie du support. Ainsi le support de la séquence $\langle\{(*,*,A)\},\{(*,Middle,*)\}\rangle$ doit être inférieur ou égal aux supports des séquences $\langle\{(*,*,A)\}\rangle$ et $\langle\{(*,Middle,*)\}\rangle$. Ainsi, pour le comptage MicroCount, le support de la séquence $\langle\{(*,*,A)\},\{(*,Middle,*)\}\rangle$ est égal à : $\frac{\min(123,120)+\min(125,115)}{742} = 0,32$.

Mise en œuvre

Ces nouveaux types de comptage du support d'une séquence multidimensionnelle peuvent s'appliquer dans n'importe quelle approche d'extraction de motifs séquentiels multidimensionnels. En effet, elle respecte bien l'antimonotonie du support, ce qui permet aux algorithmes d'extraire l'ensemble complet des séquences fréquentes. Toutefois, il est nécessaire d'adapter les algorithmes afin de permettre la recherche de la « meilleure » combinaison retrouvée dans la séquence de données d'un bloc. En effet, dans les autres approches, « la meilleure solution est la première découverte », dès que la séquence est trouvée dans le bloc, le support de la séquence est incrémenté et le calcul du support de la séquence se poursuit avec l'analyse du bloc suivant. On peut voir ces approches comme évoluant dans un contexte particulier où il n'y a pas de meilleure solution lorsqu'une séquence est supportée plusieurs fois dans un bloc, elles sont toutes équivalentes. Ainsi, chaque fois qu'une séquence est supportée par un bloc, on ajoute 1 au

support de la séquence. Dans notre contexte, si un bloc supporte une séquence, on ajoute une valeur comprise dans l'intervalle $]0, 1]$ au support global de la séquence.

Nous avons adapté l'algorithme d'extraction de motifs séquentiels multidimensionnels clos $CMSP\_Free$. Cet algorithme permet de parcourir efficacement l'espace de recherche en évitant d'extraire des connaissances redondantes. En effet, $CMSP\_Free$ extrait des motifs séquentiels multidimensionnels *clos*. Un motif multidimensionnel est clos s'il n'existe pas de séquence plus spécifique ayant le même support. Les motifs clos offrent ainsi une représentation condensée des connaissances sans perte d'information, et introduisent des propriétés efficaces d'élagage de l'espace de recherche.

$CMSP\_Free$ extrait donc les motifs clos en suivant une approche pattern-growth sans gérer d'ensemble de candidats. Ainsi, chaque fois qu'une séquence préfixe est considérée, l'algorithme vérifie s'il est possible d'intercaler un item au sein de la séquence (au début, au milieu, ou à la fin) tout en conservant le support de la séquence. Si c'est possible, alors la séquence considérée n'est pas close.

Un mécanisme similaire permet également d'élaguer efficacement l'espace de recherche en évitant d'explorer des séquences préfixes non prometteuses, c'est-à-dire des séquences s dont on est sûr qu'il n'existera pas de séquence close ayant comme préfixe une séquence s.

Experimentations

Nous avons mené des expérimentations des méthodes de comptage du support MacroCount et MicroCount appliquées à $CMSP$. Ces expérimentations ont été menées sur des données synthétiques. En effet, nous avons mené des cubes de données synthétiques. Nous considérons cinq dimensions d'analyse.

Ces expérimentations visent à étudier le nombre de motifs séquentiels multidimensionnels clos ou fréquents en fonction du seuil de support minimum ainsi que le temps d'exécution de l'extraction de tels motifs. Exprimer le nombre de motifs clos extraits en comparaison du nombre total de motifs fréquents nous permet de souligner la puissance représentative d'une telle représentation condensée. Les deux méthodes de comptages (MicroCount et MacroCount) sont étudiées.

Les courbes 3.7(a) et 3.7(c) représentent le temps d'exécution de l'extraction des motifs séquentiels multidimensionnels en fonction du seuil de support minimum considéré pour le comptage MicroCount et le comptage MacroCount. Deux jeux de données sont considérés. Le premier (Fig. 3.7(a)) est partitionné en 25 blocs alors que le second (Fig. 3.7(c)) est partitionné en 156 blocs. Du fait de l'antimonotonie du support, l'extraction des motifs est plus coûteuse dès que le support diminue. Ceci est inhérent à la problématique d'extraction de motifs. On peut remarquer qu'utiliser le comptage MacroCount est toujours plus coûteux en temps que le comptage MicroCount. En effet, même si les deux courbes suivent le même comportement, le temps d'exécution de l'extraction de motifs avec le comptage MicroCount est toujours plus rapide, dans nos expérimentations, qu'avec le comptage MacroCount.

Les courbes 3.7(b) et 3.7(d) représentent le nombre de motifs fréquents et le nombre de motifs clos extraits en fonction du seuil de support minimum considéré pour deux jeux de données différents. On

remarque que l'utilisation d'une représentation condensée nous permet d'éliminer un nombre important de connaissances redondantes pour les deux méthodes de comptage considérées. Par exemple, 26 motifs séquentiels multidimensionnels clos permettent de représenter 505 motifs séquentiels multidimensionnels pour un support à 0.11 et le comptage MicroCount utilisé dans la Fig. 3.7(b). Plus précisément, à partir de ces 26 motifs séquentiels clos, il est possible de retrouver toutes les 505 motifs séquentiels multidimensionnels avec leur support exact. Cette puissance représentative est très intéressante, car permet de ne pas présenter à l'utilisateur un ensemble de connaissances redondantes, mais un ensemble plus petit des connaissances qui permettent de retrouver les autres.

3.7 Discussion

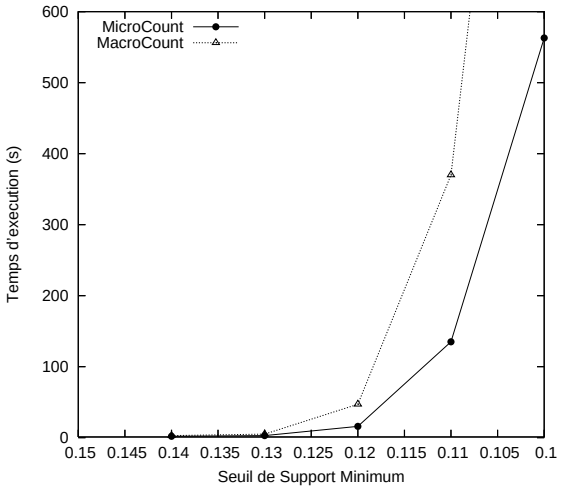
Dans ce chapitre, nous proposons trois approches différentes pour prendre en compte une dimension numérique (e.g. la mesure) dans l'extraction de motifs séquentiels multidimensionnels. L'introduction de contraintes d'agrégats permet de réduire l'espace de recherche et de ne considérer que les cellules vérifiant la condition. La discrétisation de la mesure à l'aide de partions strictes ou floues permet de prendre en compte le potentiel informationnel de la mesure en l'intégrant dans les dimensions d'analyse. Enfin, la définition de deux méthodes de comptage (macrocount et microcount) permet d'utiliser directement la mesure pour calculer le support des séquences de données multidimensionnelles. Ces trois propositions sont relativement complémentaires. L'utilisation de contraintes d'agrégats peut s'effectuer en étape de pré-traitement avant d'appliquer une des autres propositions.

De nombreuses perspectives peuvent être associées à ce travail.

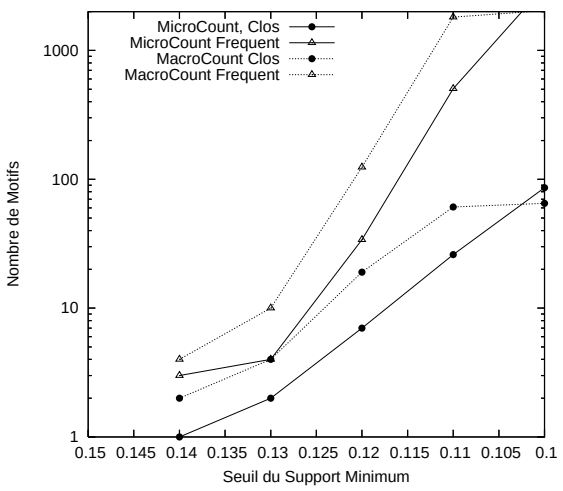
Pour l'introduction de contraintes d'agrégats, nous pouvons imaginer intégrer des contraintes plus élaborées. Il serait intéressant de combiner plusieurs contraintes (contrainte d'agrégat et contrainte sur la précision des items des séquences).

La discrétisation de la mesure est très intéressante car elle permet de conserver l'information caractérisant la cellule (l'importance de la cellule). Il est donc nécessaire de s'appuyer sur les approches de la littérature qui permettent d'établir les meilleures partitions de manières automatiques tout en restant vigilant sur la complexité de l'extraction des motifs.

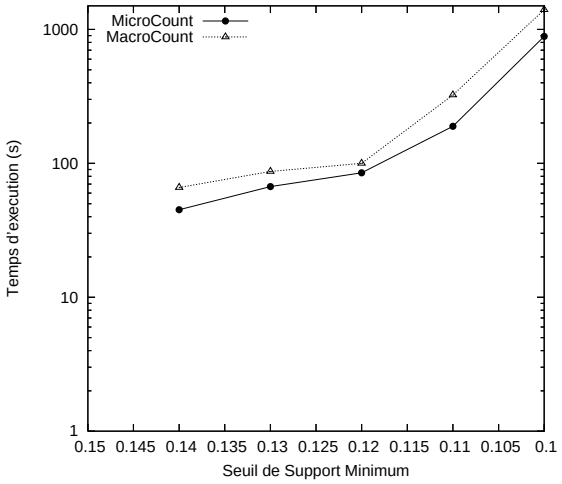
Nous pouvons imaginer utiliser les méthodes de comptage basées sur la mesure pour calculer des mesures d'intérêt basées sur le support (confiance, etc.). Il serait ainsi intéressant d'établir la confiance de règles sur des séquences en utilisant la mesure des cellules du cube qui supportent les séquences. Une telle approche pourrait permettre d'extraire des connaissances inattendues ou des exceptions dans un contexte d'extraction de motifs séquentiels multidimensionnels. Nous devons aussi étudier les cas où les dimensions ne sont pas additives avec différents opérateurs d'agrégation différents du comptage, et notamment voir les répercussions de l'utilisation de la mesure dans un tel contexte (interprétation des résultats, etc.).



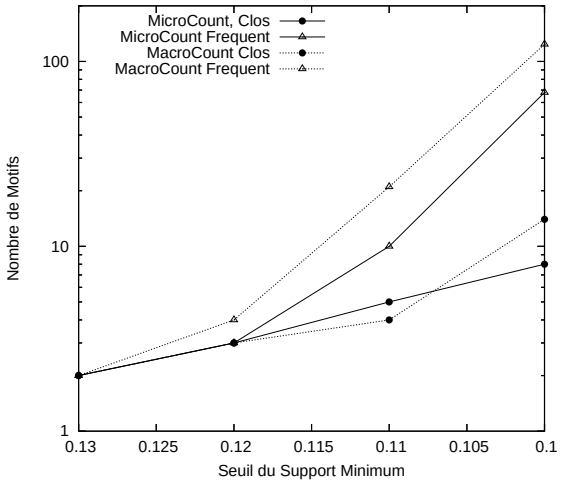
(a) Temps d'exécution en fonction du seuil de support minimum



(b) Nombre de motifs extraits en fonction du seuil de support minimum



(c) Temps d'exécution en fonction du seuil de support minimum



(d) Nombre de motifs extraits en fonction du seuil de support minimum

FIG. 3.7 – Expérimentations sur des cubes de données synthétiques

Bilan et Perspectives

Les données sont de plus en plus stockées dans des bases de données multidimensionnelles à des fins d'analyse. Elles sont donc multidimensionnelles mais également agrégées sur différents niveaux de hiérarchies. Dans la partie précédente, nous nous sommes focalisés sur la prise en compte de la multidimensionnalité. Dans cette partie, nous proposons de prendre en compte les autres spécificités des données multidimensionnelles : les *hiérarchies* et la présence d'une dimension numérique (e.g. *la mesure*).

Dans le chapitre 1, nous définissons les motifs séquentiels multidimensionnels h-généralisés qui sont décrit sur différents niveaux de hiérarchies. Nous proposons l'algorithme M^3SP afin d'extraire de tels motifs. L'algorithme M^3SP extrait des motifs séquentiels multidimensionnels h-généralisés à partir des items les plus spécifiques. Même s'il peut sembler similaire à l'algorithme M^2SP, nous montrons à l'aide d'expérimentation, que la prise en compte des hiérarchie avec l'algorithme M^2SP est beaucoup moins efficace et ne garantit pas un passage à l'échelle. D'autres expérimentations menées sur des données synthétiques et réelles montrent l'intérêt de notre proposition.

La gestion des hiérarchies permet d'améliorer la qualité des connaissances extraites. Il serait donc intéressant de proposer à l'utilisateur une gestion modulaire des hiérarchies afin d'interdire d'être l'extraction de motifs trop généraux sur certaines dimensions ou trop spécifiques sur d'autres. Nous supposons les hiérarchies définies comme des arbres. Il est nécessaire de s'intéresser au cas des hiérarchies multiples qui sont définies à l'aide de graphes orienté sans cycle. Dans de telles hiérarchies, il y a plusieurs chemins pour aller de la racine jusqu'à une feuille, ce qui peut causer une génération de candidats redondantes. Il faut donc éviter ceci afin de ne pas parcourir inutilement plusieurs fois certaines parties de l'espace de recherche. Privilégier une hiérarchie par rapport à une autre peut permettre d'élaguer plus rapidement l'espace de recherche, il peut donc être judicieux de s'intéresser à la sémantique de chaque hiérarchie et offrir à l'utilisateur la possibilité d'établir des préférences entre les hiérarchies.

Dans le chapitre 2, nous nous appuyons sur les hiérarchies pour extraire des connaissances particulières : les motifs séquentiels multidimensionnels convergents ou divergents. Les motifs convergents sont des séquences fréquentes où les items deviennent de plus en plus précis alors que dans les séquences divergentes les items deviennent de plus en plus généraux. Ce type de connaissance peut permettre d'extraire des informations intéressantes en simulant le raisonnement humain. Pour extraire ces motifs, nous nous appuyons sur un algorithme d'extraction de type *pattern growth* défini dans la partie précédente. Des expérimentations soulignent l'efficacité de cette approche.

Il est nécessaire d'effectuer une analyse plus approfondie des connaissances extraites afin de ne retenir que les séquences qui sont réellement convergentes ou divergentes et d'éviter d'extraire des séquences qui sont *artificiellement* convergentes ou divergentes. Il serait également intéressant de tester notre proposition sur d'autres types de données réelles afin d'extraire de nouveaux types de connaissances comme la naissance d'une mode (études des achats), la découverte de pandémies (données hypocratiques) ou l'extraction d'article « fondateur » (données bibliographiques). A partir de ces nouveaux types de connaissances, nous pouvons imaginer extraire les top k séquences convergentes ou divergentes.

Dans le chapitre 3, nous nous intéressons à la prise en compte d'une autre spécificité inhérente à la multidimensionnalité : la mesure. Nous montrons qu'une gestion symbolique de cette dimension « biaise » les connaissances extraites. Nous proposons trois méthodes différentes pour prendre en compte le caractère numérique de la mesure. L'introduction d'un contrainte d'agrégat permet ainsi de réduire l'espace de recherche et d'éviter les cellules dont les mesures associées sont jugées trop faibles. La discrétisation de la mesure à l'aide de partitions strictes ou floues permet d'intégrer la mesure dans les dimensions d'analyse et d'enrichir ainsi les motifs extraits par l'information présente sur cette dimension. Enfin, la mesure peut être utilisée pour calculer le support d'une séquence multidimensionnelle. Nous définissons deux méthodes de comptage différentes.

Pour l'introduction de contraintes d'agrégats, nous pouvons imaginer intégrer des contraintes plus élaborées. Il serait intéressant de combiner plusieurs contraintes (contrainte d'agrégat et contrainte sur la précision des items des séquences). La discrétisation de la mesure, est très intéressante car elle permet de conserver l'information caractérisant la cellule (l'importance de la cellule). Il est donc nécessaire de s'appuyer sur les approches de la littérature qui permettent d'établir les meilleures partitions de manière automatiques tout en restant vigilant sur la complexité de l'extraction des motifs. Nous pouvons imaginer utiliser les méthodes de comptage basées sur la mesure pour calculer des mesures d'intérêt basées sur le support (confiance, etc.). Il serait ainsi intéressant d'établir la confiance de règles sur des séquences en utilisant la mesure des cellules du cube qui supportent les séquences. Une telle approche pourrait permettre d'extraire des connaissances inattendues ou des exceptions dans un contexte d'extraction de motifs séquentiels multidimensionnels. Nous devons aussi étudier les cas où les dimensions ne sont pas additives avec différents opérateurs d'agrégation différents du comptage, et notamment voir les répercussions de l'utilisation de la mesure dans un tel contexte (interprétation des résultats, etc.).

A partir des différentes propositions développées dans les parties I et II, nous sommes capables d'extraire des connaissances dans des données complexes (multidimensionnelles, agrégées, hiérarchies) à l'aide des motifs séquentiels multidimensionnels. Ces motifs offrent une meilleure appréhension des données examinées. Ils représentent les tendances générales qui se produisent dans les données. Les motifs séquentiels multidimensionnels permettent ainsi de découvrir des relations non observées entre des événements décrits en fonction de plusieurs dimensions et plusieurs niveaux de hiérarchies. Les motifs séquentiels multidimensionnels permettent également de modéliser des comportements « généraux, classiques » à partir des données fouillées.

Extraire de tels comportements peut s'avérer fort utile dans de nombreux contextes. Toutefois, ils ne sont pas toujours suffisants. Il faut également détecter les comportements *atypiques*, c'est-à-dire les comportements qui diffèrent des autres. En effet, il peut être fort utile pour l'utilisateur d'avoir des modèles généraux sur les données, mais dans certains contextes il doit connaître les comportements qui diffèrent des autres. Par exemple, les attaques dans des systèmes informatiques ne sont pas pas des comportements généraux et doivent être détectées. Dans un contexte concurrentiel, les individus qui ne suivent pas les tendances générales sont des clients mécontents ou peut être de futurs clients. Dans tous les cas, il est nécessaire de les identifier afin de permettre à l'utilisateur de proposer une solution adaptée le plus rapidement possible.

Dans la partie suivante, nous nous attachons à extraire de tels comportements. Nous proposons deux interprétations à la notion de comportements atypiques. En effet, nous considérons qu'un comportement atypique peut être une séquence « brute » de la base de données examinée ou un motif (meta-séquence) extrait à partir de la base de données fouillée. Nous proposons une approche de détection de comportement atypique des données multidimensionnelles pour chacune de ces interprétations.

Troisième partie

Au Delà de la Fréquence : Extraction de Comportements Atypiques

A chaque fois, les exceptions l'emportent sur la règle.
John Kenneth Galbraith (1980) — *Théorie de la pauvreté de masse*

Les deux parties précédentes nous ont permis de mettre en œuvre des techniques permettant d'extraire des motifs séquentiels multidimensionnels dans des données multidimensionnelles agrégées sur plusieurs niveaux de hiérarchies. Ces motifs permettent de modéliser les comportements généraux et les tendances associées. A partir d'un motif X, on sait que $support(X)$ % des individus ou des groupes d'individus (selon B_{DB,D_R}) suivent le comportement modéliser par le motif X. L'extraction de motifs séquentiels multidimensionnels permet de faire face à de gros volumes de données en offrant à l'utilisateur une meilleure appréhension de ces données en découvrant les tendances générales, des relations non observées etc.

Néanmoins, les connaissances les plus intéressantes ne sont pas toujours celles associées aux comportements fréquents en particulier lorsque les données sont fortement corrélées. C'est ainsi que les événements rares peuvent apporter une aide non négligeable dans des contextes de prises de décisions. Il devient même primordial de les découvrir. Par exemple, un directeur marketing préférera connaître les individus qui ne suivent pas ses directives commerciales plutôt que de savoir que la quasi totalité des représentants suivent ses recommandations.

Dans cette partie, nous nous intéressons à des approches complémentaires de l'extraction de motifs séquentiels multidimensionnels vue précédemment : la recherche de comportements atypiques.

Les comportements atypiques peuvent être vus comme un ensemble de données qui se démarque sensiblement des autres données présentes dans la base. Dans ce cas là, un comportement atypique une « *donnée brute* » présente dans la base. On appelle *outliers* ou *éloignés* de tels événements [Haw80]. Il existe des approches paramétriques et des approches non paramétriques dans différents contextes (univarié, multivarié, séquences de données, cube de données).

Les comportements atypiques peuvent aussi être vus comme des connaissances sur une base qui contredisent des connaissances plus générales (support plus important) ou des croyances. L'atypicité se manifeste ainsi comme des connaissances et non comme des données particulières directement issues de la base. Selon le contexte dans lequel nous nous situons (base de croyances *a priori* ou non), ces atypicités sont caractérisées différemment. Dans le cas de base de croyances, les comportements atypiques recherchés seront des connaissances qui contredisent ces croyances. Dans le cas où aucune croyance n'est avancée sur les données, les comportements atypiques seront des connaissances qui contrediront des connaissances plus générales. Dans la littérature, on trouve fréquemment le terme de connaissances inattendues pour caractériser ces connaissances surprenantes (par rapport aux croyances ou aux connaissances plus générales).

Dans cette partie, nous traitons les deux « philosophies ». Dans le chapitre 1, nous abordons la recherche de données qui se démarquent sensiblement des autres présentes dans la base de données. Nous étudions plus précisément les approches paramétriques et non paramétriques dans différents contextes (univarié, multivarié, séquence, OLAP). Nous proposons d'extraire des séquences outiliers dans des cubes de données. Dans le chapitre 2, nous nous intéressons aux connaissances qui se démarquent soit des croyances, soit des autres connaissances plus générales dont le support est plus élevé. Nous proposons d'extraire des règles séquentielles multidimensionnelles inattendues.

Chapitre 1

Extraction de Séquences Multidimensionnelles Outliers

1.1 Introduction

Dans ce chapitre, nous abordons les comportements atypiques comme étant des « données » brutes. Nous proposons une aide à la navigation dans le cube de données par l'intermédiaire d'une recherche de séquences outliers. Nous proposons d'identifier les séquences qui se distinguent des autres à un niveau d'agrégation donné et ensuite de détecter quelles sont les causes de ces séquences outliers. Pour cela, nous nous situons à un niveau plus fin au sein de chaque séquence outlier, et nous réitérons le processus. Le processus s'arrête dès que les outliers au niveau le plus fin ont été extraits ou sinon qu'un outlier d'un niveau n'est pas outlier par rapport aux séquences communes du niveau supérieur (les autres séquences du niveau inférieur sont responsables du fait que la séquence du niveau supérieur soit outlier). Les outliers des différents niveaux identifient ainsi des chemins de navigation dans le cube de données. L'utilisateur peut ainsi naviguer pas à pas dans le cube, ou aller directement aux séquences anormales aux niveaux les plus fins.

Ce chapitre s'organise de la façon suivante. Tout d'abord, nous décrivons les travaux de la littérature qui permettent d'extraire des outliers dans différents contextes (univarié, multivarié, séquences, etc.) dans la section 1.2. Ensuite, nous exposons les motivations de notre proposition à l'aide d'un exemple d'application dans la section 1.3. Dans la section 1.4, nous détaillons notre proposition. Des expérimentations menées sur des données réelles sont rapportées et montrent l'intérêt de notre approche dans la section 1.5. Enfin, nous discutons des perspectives associées à cette proposition dans la section 1.6.

1.2 Panorama des travaux existants

Dans cette section, nous décrivons les travaux proposés pour l'extraction d'outliers tout d'abord dans un contexte classique puis dans un contexte OLAP.

Les outliers sont très répandus dans le monde réel. Les outliers peuvent avoir différentes causes. Leur atypicité peut être le résultats d'erreurs de saisie ou d'enregistrement des données mais peut également caractériser des événements réels mais rares (comportements volontairement ou involontairement non standards). Plus généralement, les outliers sont *« tellement différents des autres observations qu'ils en sont suspicieux et ont dû être générés par un autre mécanisme »* [Haw80]. Détecter des outliers est très important dans certains domaines tels que les détections de fraudes bancaires, les détections d'intrusions dans des réseaux informatiques, le suivi des performances d'athlètes, etc.

Les premiers travaux sur la détection d'outliers proviennent du monde des statistiques où de nombreuses approches ont été développées comme les tests de discordances [Haw80, BL94]. En pratique, une règle 3σ est généralement adoptée. La règle 3σ est la suivante : *Soient μ la moyenne et σ l'écart type, si une observation ne se situe pas dans l'intervalle $[\mu - 3\sigma, \mu + 3\sigma]$ alors on dit que cette observation est un outlier.* Toutes ces méthodes sont développées pour détecter un unique outlier, et ne sont plus efficaces quand plusieurs outliers sont présents dans le jeu de données. Certains suggèrent d'utiliser la médiane ou la *mad scale* au lieu de la moyenne et de l'écart type afin de détecter des outliers multiples.

Ces approches ont été développées pour extraire des outliers dans un ensemble univarié où les éléments sont supposés suivre une distribution standard (Normale, Poisson) alors que, dans de nombreux contextes applicatifs, les données sont multivariées et il est très difficile de définir la distribution qui les régit.

De nombreux travaux proposent différentes méthodes pour détecter des outliers dans des données multivariées sans connaissance a priori de la distribution. Knorr et Ng donnent leur propre définition d'outlier basée sur la distance ([KN97, KN98]). Un point est appelé outlier basé sur la distance ($db(p, D)$ outlier) si au moins une fraction p des points de l'ensemble de données sont à une distance supérieure à D. Ils prouvent aussi que leur définition est compatible avec les définitions d'outlier basées sur des distributions connues a priori ([Haw80]). Ils définissent plusieurs algorithmes pour extraire des outliers basés sur la distance.

Dans [RRS00], les auteurs montrent que les outliers basées sur la distance sont trop sensibles aux paramètres p et D. Ils définissent alors les outliers basés sur les k plus proches voisins. Ils calculent, pour chaque élément, les k plus proches voisins et ainsi la $k^{ème}$ distance. Ils ordonnent les éléments par rapport à cette distance et extraient les n plus déviants.

Dans [BKNS00], les auteurs proposent la notion d'outlier local. Ils considèrent qu'un élément est un outlier seulement quand on considère son voisinage "local". Ils assignent à chaque objet un degré qu'ils appellent facteur d'outlier local. Ainsi ils utilisent un score continu pour mesurer les outliers au lieu de donner une réponse binaire : oui ou non.

Aggarwal et Yu ([AY01]) assurent que les deux précédentes approches (distance et local) ne fonctionnent pas très bien dans des ensembles contenant de nombreuses dimensions puisque les données sont "creuses" et les outliers doivent être définis dans une projection dans un sous-espace (sub space projection). Ils proposent un algorithme évolutif permettant de détecter les outliers.

Enfin, dans ([FZFW06]), les auteurs introduisent la notion d'outlier basé sur la résolution. Ils définissent un algorithme d'extraction d'outliers qui permet d'identifier facilement les top n outliers en prenant en compte les caractéristiques locales et globales d'un ensemble de données.

[KN97, KN98] proposent une version OLAP qui permet d'extraire des cellules outliers. Sarawagi et al. [SAM98] proposent une exploration guidée par la découverte. Leur but est de découvrir des exceptions dans les cellules du cube. Ils définissent une cellule comme étant une *exception* si la mesure (agrégat) associée à la cellule diffère significativement de la valeur attendue. La valeur attendue est calculée par une formule et ils suggèrent une forme additive ou multiplicative. L'écart type peut être également estimé grâce à leur proposition. Quand la différence entre la cellule et la valeur attendue est supérieure à $2,5$ fois l'écart type, la cellule est une exception. Leur méthode peut donc être vue comme une version OLAP de la règle 3σ.

Lin et Brown ([LB03]) se focalisent aussi sur les cellules d'un cube OLAP. Ils définissent une fonction pour déterminer si la mesure d'une cellule est extrême. Quand la cellule est un outlier, les points contenus dans cette cellule sont associés. Ils combinent ainsi détection d'outlier et les concepts relatifs à OLAP afin d'établir des corrélations entre des événements (crimes).

Les méthodes précédentes ont été définies pour détecter des objets singuliers au sein de bases ou cubes de données. Elles ne permettent pas de caractériser des séquences comme outliers. Sun et al. [SCA06] proposent d'extraire des outliers dans des bases de données séquentielles. Pour approximer les mesures de distance, ils s'appuient sur des arbres probabilistes post-fixés. Toutefois cette approche se limite à des contextes où la taille du domaine de définition de la dimension étudiée est limité (4 dans leurs expérimentations).

Dans [LH07], les auteurs définissent une méthode d'extraction des comportements temporels les plus atypiques dans un cube de données. La dimension *temps* est associée à la mesure des cellules qui devient ainsi une série temporelle décrivant l'évolution de la cellule au cours du temps. Etant données une cellule « sonde » p, les auteurs proposent une méthode permettant de découvrir les k évolutions temporelles les plus atypiques parmi les descendants de la cellule sonde. Les auteurs tentent ainsi de mesurer la distance entre la série temporelle observée et la série temporelle attendue (déduite à l'aide de la cellule sonde et du poids de la cellule associée à la série temporelle). Ils identifient quatre types d'anomalies (tendance, magnitude, phase et autre, Fig. 1.1) dont la détection s'effectue par des vérifications de seuils en cascade. Toutefois, il s'avère extrêmement difficile de fixer ces seuils de manière automatique en fonction du contexte d'utilisation de cette approche.

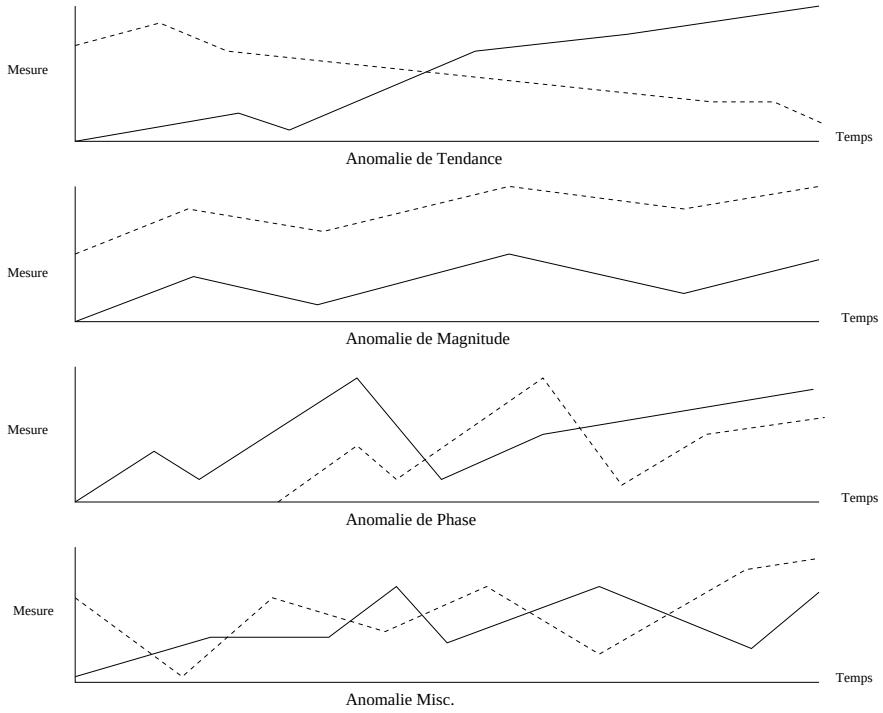


FIG. 1.1 – Les quatre types d'anomalies de [LH07]

Même si de nombreuses approches d'extraction d'outliers ont été proposées dans différents contextes, il n'existe pas d'approche permettant de caractériser des séquences outliers dans un contexte multidimensionnel (plusieurs dimensions et une mesure) où les données sont définies à différents niveaux d'agrégation.

1.3 Cube de données « exemple » et motivations

Afin d'illustrer notre proposition, nous considérons l'exemple ci-dessous tout au long de ce chapitre. Le cube de données C stocke les résultats commerciaux des diverses zones géographiques. Plus précisément, nous considérons que C est défini selon quatre dimensions comme l'indique la figure 1.3 où :

- $Date$ représente le temps (cinq dates différentes notées $1, 2, 3, 4$ et 5).
- GEO représente le lieu. Il existe une hiérarchie au sein de cette dimension qui va du niveau le plus agrégé (national) vers des niveaux de granularité plus fins (région, ville et magasin).
- $Fidélité$ représente le niveau de fidélité du client démarché. Sur ce cube, trois niveaux de fidélité sont disponibles : *gold, silver* et *platinium*.
- *Offre* représente l'option à laquelle le client a souscrit. Deux options sont présentes dans le cube de données exemple : $opt1$ et $opt2$.

Le cube de données C possède également une dimension particulière qu'est la *mesure*. La mesure est une application de : $Dom(GEO) \times Dom(date) \times Dom(Fidélité) \times Dom(Offre) \rightarrow Dom(Mesure)$. La mesure représente pour un lieu, le nombre d'options souscrites à une date donnée par des clients selon leur degré de fidélité.

Par exemple, le premier 5-uplet $\langle(Paris, 1, Gold, opt1) : 800\rangle$ de la figure 1.3 indique que 800 clients $Gold$ à $Paris$, à la date 1, ont souscrit à l'option $opt1$. La figure 1.2 représente le cube de données exemple pour la date 1.

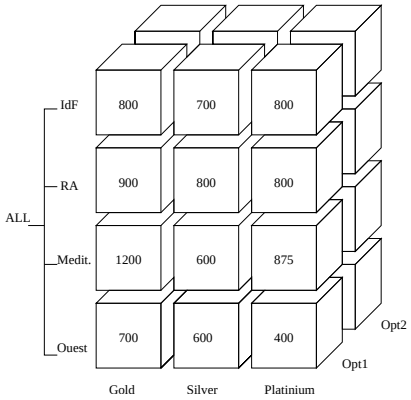


FIG. 1.2 – Cube de données Exemple pour la date 1

Nous souhaitons proposer à l'utilisateur un nouveau mode de navigation dans un cube de données. Cette navigation se base sur la recherche de séquences outliers. Nous proposons ainsi d'identifier à chaque niveau les n séquences qui diffèrent le plus des autres et ensuite de réitérer ce processus sur les séquences outliers à des niveaux inférieurs.

Ainsi, dans le cube de données exemple, *GEO Sud* ne suit pas le même comportement que les autres positions sur GEO (Paris, Ouest, RA). Nous allons donc nous repositionner à un niveau de granularité plus fin dans ce sous-cube afin d'essayer d'identifier les raisons pour lesquelles le sous-cube identifié par *GEO Sud* est outlier, et ainsi extraire de nouveaux outliers dans ce sous-cube (Drill down sur GEO= Sud). Si les outliers extraits dans ce sous-cube ne suivent pas le même comportement que les séquences "communes" du niveau supérieur, alors ces séquences peuvent être considérées comme responsables du fait que la séquence *Sud* soit outlier, on parle de *séquences totalement outliers*. Si une séquence outlier d'un niveau suit le même comportement que les séquences « communes » du niveau supérieur, on parle alors de *séquences localement outliers.* On peut donc réitérer le processus à des niveaux d'agrégation plus fins sur les nouvelles séquences totalement outliers, les outliers locaux ne pouvant pas être considérés comme cause de la « rareté » de la séquence de niveau supérieur.

Nous pouvons imaginer nous situer dans un contexte où l'utilisateur est un directeur marketing national. Il veut vérifier dans un premier temps si tous les centres inter-régionnaux suivent ses directives. Si un centre ne respecte pas ses directives, il veut voir si c'est au niveau du centre que les recommandations ne sont pas suivies ou si ce sont des sous-centres qui ne respectent pas ses indications et nuisent ainsi à la production du centre inter-régionnal.

GEO	Date	Fidélité	Offre	Mesure
Paris	1	Gold	opt1	800
Paris	1	Gold	opt2	1000
Paris	1	Silver	opt1	700
Paris	1	Silver	opt2	700
Paris	1	Platinium	opt1	800
Paris	1	Platinium	opt2	900
Paris	2	Gold	opt1	1000
Paris	2	Gold	opt2	900
Paris	2	Silver	opt1	800
Paris	2	Silver	opt2	900
Paris	2	Platinium	opt1	900
Paris	2	Platinium	opt2	1300
Paris	3	Gold	opt1	1200
Paris	3	Gold	opt2	750
Paris	3	Silver	opt1	750
Paris	3	Silver	opt2	1000
Paris	3	Platinium	opt1	1300
Paris	3	Platinium	opt2	1000
Paris	4	Gold	opt1	1400
Paris	4	Gold	opt2	500
Paris	4	Silver	opt1	800
Paris	4	Silver	opt2	1200
Paris	4	Platinium	opt1	900
Paris	4	Platinium	opt2	1050
Paris	5	Gold	opt1	1500
Paris	5	Gold	opt2	500
Paris	5	Silver	opt1	690
Paris	5	Silver	opt2	1200
Paris	5	Platinium	opt1	850
Paris	5	Platinium	opt2	1100

(a) GEO Paris

GEO	Date	Fidélité	Offre	Mesure
RA	1	Gold	opt1	900
RA	1	Gold	opt2	1010
RA	1	Silver	opt1	800
RA	1	Silver	opt2	650
RA	1	Platinium	opt1	800
RA	1	Platinium	opt2	750
RA	2	Gold	opt1	1095
RA	2	Gold	opt2	910
RA	2	Silver	opt1	810
RA	2	Silver	opt2	870
RA	2	Platinium	opt1	900
RA	2	Platinium	opt2	1220
RA	3	Gold	opt1	1270
RA	3	Gold	opt2	730
RA	3	Silver	opt1	805
RA	3	Silver	opt2	1100
RA	3	Platinium	opt1	1300
RA	3	Platinium	opt2	1050
RA	4	Gold	opt1	1440
RA	4	Gold	opt2	580
RA	4	Silver	opt1	795
RA	4	Silver	opt2	1230
RA	4	Platinium	opt1	900
RA	4	Platinium	opt2	1070
RA	5	Gold	opt1	1490
RA	5	Gold	opt2	540
RA	5	Silver	opt1	720
RA	5	Silver	opt2	1220
RA	5	Platinium	opt1	850
RA	5	Platinium	opt2	1090

(b) GEO RA

GEO	Date	Fidélité	Offre	Mesure
Sud	1	Gold	opt1	1200
Sud	1	Gold	opt2	1300
Sud	1	Silver	opt1	600
Sud	1	Silver	opt2	750
Sud	1	Platinium	opt1	875
Sud	1	Platinium	opt2	850
Sud	2	Gold	opt1	1250
Sud	2	Gold	opt2	800
Sud	2	Silver	opt1	700
Sud	2	Silver	opt2	900
Sud	2	Platinium	opt1	910
Sud	2	Platinium	opt2	900
Sud	3	Gold	opt1	1160
Sud	3	Gold	opt2	950
Sud	3	Silver	opt1	550
Sud	3	Silver	opt2	1000
Sud	3	Platinium	opt1	975
Sud	3	Platinium	opt2	940
Sud	4	Gold	opt1	1080
Sud	4	Gold	opt2	1000
Sud	4	Silver	opt1	700
Sud	4	Silver	opt2	800
Sud	4	Platinium	opt1	950
Sud	4	Platinium	opt2	1400
Sud	5	Gold	opt1	1100
Sud	5	Gold	opt2	650
Sud	5	Silver	opt1	690
Sud	5	Silver	opt2	750
Sud	5	Platinium	opt1	880
Sud	5	Platinium	opt2	1000

(c) GEO Sud

GEO	Date	Fidélité	Offre	Mesure
Ouest	1	Gold	opt1	700
Ouest	1	Gold	opt2	870
Ouest	1	Silver	opt1	600
Ouest	1	Silver	opt2	500
Ouest	1	Platinium	opt1	400
Ouest	1	Platinium	opt2	800
Ouest	2	Gold	opt1	750
Ouest	2	Gold	opt2	800
Ouest	2	Silver	opt1	690
Ouest	2	Silver	opt2	745
Ouest	2	Platinium	opt1	600
Ouest	2	Platinium	opt2	1270
Ouest	3	Gold	opt1	900
Ouest	3	Gold	opt2	720
Ouest	3	Silver	opt1	740
Ouest	3	Silver	opt2	1050
Ouest	3	Platinium	opt1	1100
Ouest	3	Platinium	opt2	1050
Ouest	4	Gold	opt1	1200
Ouest	4	Gold	opt2	450
Ouest	4	Silver	opt1	810
Ouest	4	Silver	opt2	1150
Ouest	4	Platinium	opt1	700
Ouest	4	Platinium	opt2	1000
Ouest	5	Gold	opt1	1470
Ouest	5	Gold	opt2	460
Ouest	5	Silver	opt1	750
Ouest	5	Silver	opt2	1230
Ouest	5	Platinium	opt1	650
Ouest	5	Platinium	opt2	1060

(d) GEO Ouest

FIG. 1.3 – Cube de données Exemple sous forme tabulaire

1.4 Proposition d'une recherche guidée de séquences rares

Dans cette section, vous présentons notre contribution. Tout d'abord, nous rappelons les particularités des données que nous allons manipuler. Ensuite nous verrons comment nous mesurons d'une part, la distance entre deux séquences et d'autre part, entre une séquence et un ensemble de séquences. Enfin, nous présenterons les algorithmes permettant la recherche guidée d'outliers.

1.4.1 Données manipulées

Comme dans les chapitres précédents, nous supposons que parmi toutes les dimensions définissant un cube de données, il existe au moins une dimensions dont le domaine est ordonné (e.g. dimension temporelle). Pour tout cube défini sur les dimensions $\mathcal{D}$, nous appliquons la partition de $\mathcal{D}$ définie précédemment.

Il en découle que chaque cellule $cell = \langle (d_1, \ldots, d_n) : \mu \rangle$ d'un cube peut être notée $cell = \langle (i, r, a, t) : \mu \rangle$ où i, r, a et t correspondent respectivement aux restrictions de $cell$ sur D_I, D_R, D_A et D_T.

Dans le cadre de l'extraction de séquences outliers, l'ensemble D_R permet d'identifier les sous-cubes par rapport auxquels les séquences anormales seront extraites. Pour cette raison, cet ensemble est nommé *référence*. Chaque n-uplet défini sur D_R identifie une séquence. Nous désirons rechercher les séquences identifiées par des sous-cubes qui dévient fortement des autres.

L'ensemble D_T permet d'introduire une relation d'ordre entre les cellules. Cet ensemble permet donc d'introduire la notion de séquentialité.

L'ensemble D_A décrit les axes d'*analyse*, c'est-à-dire l'ensemble des dimensions apparaissant explicitement dans les séquences extraites.

Il est aussi possible de définir un sous-ensemble D_I qui permet de décrire les axes *ignorés*, c'est-à-dire les dimensions qui ne servent ni à introduire une relation d'ordre, ni identifier un sous-cube, ni à définir la séquence elle-même. Ces dimensions dites *oubliées* peuvent être vues comme instanciées avec la valeur ALL.

Par rapport au cube de données exemple, nous choisissons :

- $D_T = \{date\}$
- $D_A = \{Fidélité, Offre\}$
- $D_R = \{GEO\}$
- $D_I = \{\}$

On note C_{D_R} l'ensemble des sous-cubes à partir des dimensions de référence. Ainsi, la figure 1.3 montre les quatre sous-cubes définis en fonction de $D_R = \{GEO\}$, c'est-à-dire C_{D_R}.

On note également $C_{(d_{r_1}, d_{r_2}, \ldots, d_{r_k})}$, le sous-cube identifié par le k-uplet $(d_{r_1}, d_{r_2}, \ldots, d_{r_k})$ défini sur les dimensions de références. Conformément au cube exemple, la figure 1.3(a) représente $C_{(Paris)}$.

1.4.2 Blocs et séquences

Définition 1.1 (t-Bloc). On appelle t-bloc B un ensemble de cellules dont les positions sur D_R et D_T sont fixes. B est l'ensemble des n-uplets prenant leurs valeurs sur D_A :

$$B = \{\langle ({d_{i1}}^1, \ldots, {d_{im}}^1) : {\mu^1}\rangle, \ldots, \langle ({d_{i1}}^p, \ldots, {d_{im}}^p) : {\mu^p}\rangle\}$$

On notera $B = \{c_1, \ldots, c_p\}$.

Définition 1.2 (Séquence). On appelle séquence, une liste ordonnée non vide de t-blocs de la forme :

$\varsigma = \langle\{\langle ({d_{i1}}^1, \ldots, {d_{im}}^1) : {\mu^1}\rangle, \ldots, \langle ({d_{i1}}^p, \ldots, {d_{im}}^p) : {\mu^p}\rangle\}, \ldots \{\langle ({d_{i1}}^{1'}, \ldots, {d_{im}}^{1'}) : {\mu^{1'}}\rangle, \ldots, \langle ({d_{i1}}^{p'}, \ldots, {d_{im}}^{p'}) : {\mu^{p'}}\rangle\}\rangle$

On note $\varsigma = \langle B_1, \ldots, B_l\rangle$.

Etant donnée la partition de l'ensemble de dimensions (D_A, D_R, D_t, d_F), un cube de données peut être vu comme un ensemble de séquences. Chaque séquence est identifiée par une valeur prise sur D_R. Chaque t-bloc d'une séquence correspond à une valeur sur D_T. Les t-blocs prennent ainsi leurs valeurs sur D_A pour des valeurs fixes de D_R et D_T.

La figure 1.4 illustre une séquence où $D_A = \{$A_1,A_2$\}$ pour une position sur D_R fixée. La séquence regroupe en fonction d'un ordre relatif à D_T les t-blocs dont les positions des cellules varient uniquement sur D_A. La séquence représentée dans la figure 1.4 contient quatre t-blocs différents. Les cellules colorées correspondent aux cellules non vides.

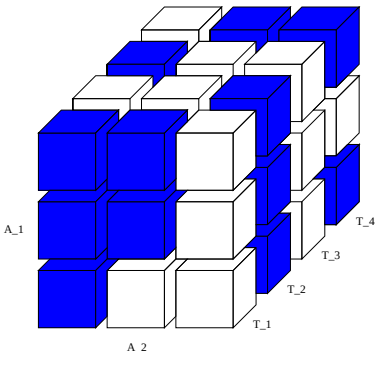


FIG. 1.4 – Séquence de blocs pour une valeur de D_R

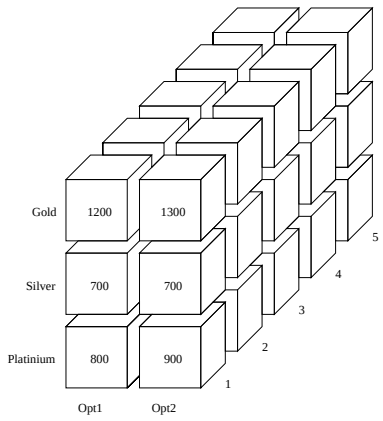


FIG. 1.5 – Séquence pour $GEO = Sud$

La figure 1.5 représente la séquence identifiée par $GEO = Sud$. Nous avons deux dimensions d'analyse. La séquence contient cinq blocs (les blocs aux dates $1, 2, 3, 4$ et 5).

1.4.3 Comparaison de séquences

Afin de définir si une séquence est outlier ou non, il est nécessaire de pouvoir calculer la similarité entre cette séquence et les autres. Nous introduisons ainsi une mesure de comparaison entre deux séquences. Cette mesure peut être une distance ou une mesure de similarité. Si la distance entre deux séquences est grande alors ces deux séquences seront considérées comme éloignées. De façon inverse, une mesure de similarité essaie de voir à quel point deux séquences sont proches. Si cette valeur est 1 alors ces séquences sont considérées comme identiques et si la mesure est égale à 0, alors les deux séquences n'ont rien en commun.

La distance la plus connue entre deux séquences est la *distance de Levenshtein ou d'édition* [Lev66]. La distance de Levenshtein entre deux séquences s_1 et s_2 correspond aux nombres d'opérations d'édition (insertion, suppression, déplacement) nécessaires pour transformer la séquence s_1 en la séquence s_2. Dans notre contexte, cette mesure n'est pas suffisante. En effet, la distance d'édition de deux séquences peut être très faible (1 opération) alors que les séquences sont en opposition de phase. Imaginons un centre qui suit les directives nationales, et qui a un comportement périodique. Un autre centre est en total décalage avec les directives nationale et se retrouve ainsi décalé d'une demi-période par rapport à la séquence précédente. La distance d'édition entre ces deux séquences est faible et ne traduit pas ce décalage.

Nous nous basons ici sur les distances les plus classiquement utilisées : la distance euclidienne, la distance de Manhattan et une mesure de similarité basée sur le cosinus.

Pour pouvoir établir des distances ou des mesures de similarité entre deux séquences, nous introduisons la notion de cellules comparables.

Définition 1.3 (Cellules comparables)**.** Deux cellules $c_1 = \langle (d_1, \ldots, d_n), \mu \rangle$ et $c_2 = \langle (d'_1, \ldots, d'_n), \mu' \rangle$ sont comparables si leurs restrictions sur D_A sont égales : $c_1.D_A = c_2.D_A$.

Par exemple, les cellules $c_1 = \langle (Ouest, 1, Gold, opt1), 1200 \rangle$ et $c_2 = \langle (RA, 1, Gold, opt1), 900 \rangle$ sont comparables puisqu'elles ont la même restriction $(Gold, opt1)$ sur D_A. Par contre, les cellules c_1 et $c_3 = \langle (Sud, 1, Silver, opt2), 600 \rangle$ sont incomparables étant donné que leurs restrictions sur D_A sont différentes.

Dans ce chapitre, nous nous situons dans un contexte de cube de données dense et nous supposons qu'il existe très peu de cellules vides. Pour calculer la distance entre deux blocs, nous essayons de construire des vecteurs de mesure, où chaque dimension sur les deux vecteurs correspond à des valeurs de mesures entre deux cellules comparables. L'algorithme 15 décrit comment deux blocs sont transformés en deux vecteurs contenant les valeurs de mesures des cellules comparables.

La représentation vectorielle de deux blocs permet d'appliquer les mesures de distances et de similarités telles que la distance euclidienne et le cosinus. Le calcul de la distance entre deux blocs nous permet de calculer la distance entre deux séquences :

Algorithme 15 : TransBlocVec : Construction des vecteurs représentant les blocs

Entrées : b_1 et b_2 blocs

Sorties : Construction de deux vecteurs v_1 et v_2

début

$v_1 \leftarrow ()$

$v_2 \leftarrow ()$

pour chaque *cellule* $c_i \in b_1$ **faire**

si $\exists c_j \in b_2 \mid c_i$ *et* c_j *sont comparables* **alors**

$v_1.add(mesure(c_i))$

$v_2.add(mesure(c_j))$

retourner v_1, v_2

fin

Définition 1.4 (Distance entre 2 séquences). Soient $s_1 = \langle b_1, b_2, \ldots, b_k \rangle$ et $s_2 = \langle b'_1, b'_2, \ldots, b'_k \rangle$ deux séquences multidimensionnelles, $dist$ une mesure de distance et Op un opérateur d'agrégation. La distance entre s_1 et s_2 se définit de la façon suivante :

$d(s_1, s_2) = Op(dist(v_j, v'_j)$ pour $j = 1 \ldots k$

Dans ce chapitre, nous utilisons les distances de Manhattan et euclidienne définies ci-dessous. Nous utilisons aussi la mesure de similarité basée sur le cosinus.

Distance de Manhattan :

$$Man(v_1, v_2) = \sum_{k=0}^{m} |v_{1_k} - v_{2_k}|$$

Distance euclidienne :

$$Euclid(v_1, v_2) = \sqrt{\sum_{k=0}^{m} (v_{1_k} - v_{2_k})^2}$$

Cosinus :

$$cos(v_1, v_2) = \frac{v_1.v_2}{||v_1||||v_2||} = \frac{\sum_{k=0}^{m}(v_{1_k} v_{2_k})}{\sqrt{\sum_{k=0}^{m} v_{1_k}^2 \sum_{k=0}^{m} v_{2_k}^2}}$$

La définition 1.4 est suffisament générique pour appliquer n'importe quel opérateur d'agrégation pour calculer une distance entre deux séquences. La distance entre deux séquences peut être, par exemple, pour une mesure de distance donnée, la moyenne des distances entre chaque t-blocs, la médiane ou le max.

1.4.4 Comparaison de séquence par rapport à un ensemble de séquences

Pour déterminer si une séquence est un outlier, il est nécessaire de connaître sa similarité par rapport à toutes les autres séquences de la base. Nous établissons donc une matrice de distance représentant les distance entre chaque séquence de la base.

Sequence_Id	1	2	...	l
1	1	$sim(1,2)$	...	$sim(1,l)$
2	•	1	...	$sim(2,l)$
...	•	•	1	...
l	•	•	•	1

(a) Matrice de similarité

Sequence_Id	1	2	...	l
1	0	$d(1,2)$	...	$d(1,l)$
2	•	0	...	$d2,l)$
...	•	•	0	...
l	•	•	•	0

(b) Matrice de distance

FIG. 1.6 – Comparaison d'une séquence par rapport aux autres

Nous définissons la distance (resp. la similarité) d'une séquence par rapport à un ensemble de séquences, comme la moyenne des distances (resp. similarités) entre la séquence et les autres séquences. La distance d'une séquence s_α par rapport à un ensemble de séquences S est couramment définie dans la littérature de la façon suivante :

$$d(s_\alpha, S) = \frac{\Sigma_{i=1}^{i<\alpha} d(s_\alpha, s_i) + \Sigma_{j=\alpha+1}^{|s|} d(s_j, s_\alpha)}{|S| - 1}$$

Le calcul de la distance d'une séquence par rapport à un ensemble de séquences est primordial pour savoir si une séquence est un outlier ou non. Il est possible de définir un outlier par rapport à un seuil de distance fixé a priori par l'utlisateur. Définir ce seuil est très fastidieux et dépend fortement des séquences examinées. Il est, en conséquent, plus aisé pour l'utilisateur de définir un entier k qui correspond aux nombres d'outliers qu'il souhaite avoir. L'utilisateur veut voir ainsi les k séquences qui diffèrent le plus des autres.

Définition 1.5 (top n outliers)**.** Une séquence s_α est un top n outlier s'il n'existe pas plus de $n-1$ séquences telles que $d(s_i, C_{D_R}) > d(s_\alpha, C_{D_R})$

Exemple 1.1 (Top 1 outlier)**.** Etant donné le cube exemple (figure 1.3), nous voulons identifier la séquence la plus outlier. Il est nécessaire de calculer la matrice de distance entre les différentes séquences. La figure 1.7 (a) représente la matrice en fonction de la distance de Manhattan (moyenne). La figure 1.7 (b) représente la matrice de distance en fonction de la distance Euclidienne.

Les figures 1.7 (c) et 1.7 (d) représentent la distance moyenne d'une séquence par rapport aux autres en fonction de la mesure de distance utilisée.

Pour les deux types de distance, la séquence identifiée par $GEO = Sud$ est considérée comme top 1 outlier puisque c'est la séquence la plus éloignée des autres.

1.4.5 Algorithmes

Il s'agit ici de fournir les méthodes et outils à l'utilisateur pour qu'il soit capable, face à une séquence identifiée comme un outlier à un haut niveau de granularité, d'étudier plus en détail les sous-données

Sequence_Id	$Paris$	RA	Sud	$Ouest$
$Paris$	0	243	1102	715
RA	•	0	1145	798
Sud	•	•	0	1437
$Ouest$	•	•	•	0

(a) Distance de Manhattan (moyenne)

Sequence_Id	$Paris$	RA	Sud	$Ouest$
$Paris$	0	127	576	365
RA	•	0	558	408
Sud	•	•	0	699
$Ouest$	•	•	•	0

(b) Distance Euclidienne (moyenne)

Sequence_Id	$Paris$	RA	Sud	$Ouest$
Distance	686	728	1228	983

(c) Distance par rapport à l'ensemble (Manhattan)

Sequence_Id	$Paris$	RA	Sud	$Ouest$
Distance	356	364	611	491

(d) Distance par rapport à l'ensemble (Euclidienne)

FIG. 1.7 – Comparaison d'une séquence par rapport aux autres dans le cube exemple

associées à un niveau plus fin. Cette méthodologie permet de le guider dans sa recherche afin qu'il cible le plus directement possible les données susceptibles de l'intéresser.

Dans cette section, on notera :

– $S_{v_{R_i}}$ la séquence identifiée par $D_R = v_{R_i}$;

– $C_{D_R=v_R}$ le sous-cube relatif à v_R.

Chaque valeur v_R sur D_R identifie une séquence. Ainsi si une séquence est un top n outlier, alors ce sont les actions sur v_R qui sont anormales par rapport aux actions relatives aux autres valeurs sur D_R. Comme v_R n'est pas le niveau le plus fin dans la hiérarchie, il est toujours possible de se demander pourquoi v_R est outlier. Nous pouvons donc nous placer dans le sous-cube identifié par v_R et rechercher les top n outliers.

L'algorithme 16 permet d'extraire les top n outliers à un niveau d'agrégation donnée. Pour chaque séquence top n outlier identifiée par sa valeur v_R sur D_R, le processus est réitéré sur les sous-cubes identifiés par chaque valeur v_R jusqu'à arriver au niveau d'agrégation le plus fin.

Algorithme 16 : RechTopn

Entrées : C_{v_R} Cube de données, n entier, L ensemble, $dist$ une mesure de distance

Sorties : Séquences outliers à chaque niveau de granularité

début

  Calculer la matrice de distance

  pour chaque *séquence* $S_{v_{R_i}} \in C_{v_R}$ *top* n *outlier* **faire**

    $add(v_{R_i}, L)$

    si v_{R_i} *is not* $leaf$ **alors**

      $RechTopn(C_{DrillDown(v_{R_i})}, n, L, dist)$

  retourner v_1, v_2

fin

Pour $k = 1$, cet algorithme permet de proposer à l'utilisateur un chemin de navigation dans le cube afin d'identifier des séquences anormales par rapport à l'ensemble des données. Pour $k = 1$, le chemin regroupe les valeurs v_R dont les séquences associées sont des top 1 outliers à un niveau donné. Ce chemin part d'un niveau d'agrégation élevé et se termine au niveau d'agrégation le plus fin. Grâce à ce chemin, l'utilisateur peut directement aller sur la valeur v_R la plus fine, ou avancer pas à pas.

Pour $k \geq 1$, l'algorithme propose un arbre de navigation. En effet, il n'existe plus un seul chemin, mais plusieurs chemin. L'utilisateur peut ainsi visualiser les séquences anormales par l'intermédiaire de cet arbre. Il peut directement situer au niveau d'agrégation le plus fin (les feuilles), ou naviguer à travers les différents nœuds de l'arbre.

Une séquence peut être un top n outlier pour plusieurs raisons :

- Une séquence à un niveau inférieur est sensiblement différente des autres. La séquence a ainsi une importance dans le fait que la séquence agrégée du niveau supérieur est outlier. Dans ce cas là, l'algorithme 16 permet d'extraire ces différents outliers pour chaque niveau.
- Une grande partie des séquences du niveau inférieur sont sensiblement différentes du comportement général du niveau supérieur des séquences non outliers. Ainsi, une séquence qui suit le comportement général peut être considérée comme un top n outlier. Nous proposons donc de calculer la distance de cette séquence avec les autres séquences non outliers afin de voir si cette séquence suit le comportement général (bien le seul). Comme nous ne nous situons pas au même niveau d'agrégation, il est nécessaire de normaliser les séquences afin de calculer la distance entre deux séquences de niveaux d'agrégation différents.

Nous parlerons donc de séquences *totalement outliers* pour les séquences qui se démarquent des comportements généraux à la fois au niveau L et au niveau supérieur $L - 1$. Dans le cas contraire, nous parlerons de séquences *localement outliers* pour qualifier les séquences outliers au niveau L mais *normales* au niveau $L - 1$. Notons que les termes d'outliers locaux et globaux diffèrent de la définition de [BKNS00].

Nous pouvons adapter l'algorithme 16 afin d'arrêter la construction de chemins quand on arrive sur les niveaux les plus fins ou quand les outliers d'un niveau suivent le comportement général du niveau supérieur. L'algorithme 17 prend en compte ce type de navigation et inclut donc la comparaison avec le comportement au niveau supérieur.

Exemple : Naviguons à travers le cube exemple, à l'aide de l'extraction des top 1 outliers. Dans un premier temps, nous nous fixons au niveau le plus agrégé de la dimension de référence, c'est-à-dire, les fils de la racine (Paris, Ouest, RA, Sud). La séquence identifiée par *Sud* est un top 1 outlier. Nous nous situons donc sur les fils de *Sud*, c'est-à-dire les séquences identifiées par *Nice, Perpigan, Marseille* et *Montpellier* comme indiqué par la figure 1.8.

La séquence identifiée par *Montpellier* est un top 1 outlier. Nous vérifions si cette séquence est outlier par rapport aux séquences *normales* du niveau supérieur (Paris, RA, Ouest) normalisées. Cette

Algorithme 17 : RechTopnUp

Entrées : C_{v_R} Cube de données, n entier, L ensemble, $dist$ une mesure de distance

Sorties : Séquences outliers à chaque niveau de granularité bis

début

Calculer la matrice de distance

pour chaque *séquence* $S_{v_{R_i}} \in C_{v_R}$ *top* n *outlier* **faire**

$add(v_{R_i}, L)$

si v_{R_i} *is not leaf* $\wedge\, Normal(S_{v_{R_i}})$ *is not top* n *outlier in* $C_{roll\ up(v_R)}$ **alors**

$RechTopn(C_{DrillDown(v_{R_i})}, n, L, dist)$

retourner v_1, v_2

fin

séquence est également un top 1 outlier dans ce contexte normalisé. Nous nous situons au niveau le plus fin, l'algorithme s'arrête donc.

1.5 Expérimentations

Nous avons effectué des expérimentations sur un cube de données d'EDF. Le cube se compose d'une dimension géographique choisie comme dimension de référence, d'une dimension temporelle, et quatre dimensions d'analyse. Le cube au niveau d'agrégation le plus élevé contient 35000 cellules organisées en 8 séquences de blocs. L'opérateur d'agrégation utilisé sur la mesure est un comptage.

Les expérimentations rapportées dans cette section montrent le temps d'exécution et le nombre d'outliers extraits en fonction du nombre de top k outliers recherchés au niveau d'agrégation le plus élevé. Nous étudions ensuite, de manière plus précise, les outliers extraits. Nous regardons s'ils se démarquent uniquement de leur sous-cube ou s'ils se démarquent aussi du comportement général du niveau supérieur. Ces expérimentations sont menées avec trois mesures différentes (distance euclidienne, distance de Manhattan, et mesure de similarité cosinus) et rois topérateurs d'agrégation différents (moyenne, médiane, et min).

Les figures 1.9(a) et 1.9(b) montrent respectivement le temps d'exécution et le nombre d'outliers extraits en fonction du nombre de top k outliers recherchés. Le temps d'exécution et le nombre d'outliers augmentent proportionnellement avec le paramètre k.

Les figures restantes montrent le nombre de séquences outliers qui sont "totalement" outliers pour différents opérateurs d'agrégation (Fig 1.9(c), 1.10(a), et 1.10(c)), et le nombre de séquences outliers, pour différents opérateurs d'agrégation (Fig 1.9(d), 1.10(b), et 1.10(d)), qui suivent le comportement général du niveau supérieur par rapport au paramètre k. Quels que soient la mesure et l'opérateur d'agrégation utilisés, le nombre de séquences qui sont outliers par rapport à leur sous-cube et par rapport aux séquences du niveau supérieur augmente avec le paramètre k. Pour les séquences qui sont outliers dans

GEO	Date	Fidélité	Offre	Mesure
Nice	1	Gold	opt1	200
Nice	1	Gold	opt2	250
Nice	1	Silver	opt1	150
Nice	1	Silver	opt2	175
Nice	1	Platinium	opt1	200
Nice	1	Platinium	opt2	225
Nice	2	Gold	opt1	250
Nice	2	Gold	opt2	225
Nice	2	Silver	opt1	250
Nice	2	Silver	opt2	225
Nice	2	Platinium	opt1	225
Nice	2	Platinium	opt2	300
Nice	3	Gold	opt1	300
Nice	3	Gold	opt2	187.5
Nice	3	Silver	opt1	180
Nice	3	Silver	opt2	250
Nice	3	Platinium	opt1	325
Nice	3	Platinium	opt2	250
Nice	4	Gold	opt1	350
Nice	4	Gold	opt2	125
Nice	4	Silver	opt1	200
Nice	4	Silver	opt2	300
Nice	4	Platinium	opt1	225
Nice	4	Platinium	opt2	275
Nice	5	Gold	opt1	375
Nice	5	Gold	opt2	125
Nice	5	Silver	opt1	172.5
Nice	5	Silver	opt2	300
Nice	5	Platinium	opt1	212.5
Nice	5	Platinium	opt2	275

(a) GEO Nice

GEO	Date	Fidélité	Offre	Mesure
Perpignan	1	Gold	opt1	210
Perpignan	1	Gold	opt2	300
Perpignan	1	Silver	opt1	130
Perpignan	1	Silver	opt2	165
Perpignan	1	Platinium	opt1	210
Perpignan	1	Platinium	opt2	220
Perpignan	2	Gold	opt1	270
Perpignan	2	Gold	opt2	215
Perpignan	2	Silver	opt1	200
Perpignan	2	Silver	opt2	225
Perpignan	2	Platinium	opt1	215
Perpignan	2	Platinium	opt2	275
Perpignan	3	Gold	opt1	285
Perpignan	3	Gold	opt2	197.5
Perpignan	3	Silver	opt1	190
Perpignan	3	Silver	opt2	235
Perpignan	3	Platinium	opt1	335
Perpignan	3	Platinium	opt2	245
Perpignan	4	Gold	opt1	320
Perpignan	4	Gold	opt2	150
Perpignan	4	Silver	opt1	195
Perpignan	4	Silver	opt2	250
Perpignan	4	Platinium	opt1	235
Perpignan	4	Platinium	opt2	270
Perpignan	5	Gold	opt1	350
Perpignan	5	Gold	opt2	145
Perpignan	5	Silver	opt1	160.5
Perpignan	5	Silver	opt2	245
Perpignan	5	Platinium	opt1	215.5
Perpignan	5	Platinium	opt2	275

(b) GEO Perpignan

GEO	Date	Fidélité	Offre	Mesure
Marseille	1	Gold	opt1	210
Marseille	1	Gold	opt2	300
Marseille	1	Silver	opt1	200
Marseille	1	Silver	opt2	185
Marseille	1	Platinium	opt1	195
Marseille	1	Platinium	opt2	230
Marseille	2	Gold	opt1	230
Marseille	2	Gold	opt2	220
Marseille	2	Silver	opt1	175
Marseille	2	Silver	opt2	225
Marseille	2	Platinium	opt1	220
Marseille	2	Platinium	opt2	305
Marseille	3	Gold	opt1	315
Marseille	3	Gold	opt2	177.5
Marseille	3	Silver	opt1	187.5
Marseille	3	Silver	opt2	245
Marseille	3	Platinium	opt1	315
Marseille	3	Platinium	opt2	255
Marseille	4	Gold	opt1	330
Marseille	4	Gold	opt2	175
Marseille	4	Silver	opt1	210
Marseille	4	Silver	opt2	250
Marseille	4	Platinium	opt1	215
Marseille	4	Platinium	opt2	280
Marseille	5	Gold	opt1	350
Marseille	5	Gold	opt2	125
Marseille	5	Silver	opt1	180
Marseille	5	Silver	opt2	200
Marseille	5	Platinium	opt1	205
Marseille	5	Platinium	opt2	285

(c) GEO Marseille

GEO	Date	Fidélité	Offre	Mesure
Montpellier	1	Gold	opt1	580
Montpellier	1	Gold	opt2	450
Montpellier	1	Silver	opt1	120
Montpellier	1	Silver	opt2	225
Montpellier	1	Platinium	opt1	270
Montpellier	1	Platinium	opt2	125
Montpellier	2	Gold	opt1	500
Montpellier	2	Gold	opt2	140
Montpellier	2	Silver	opt1	75
Montpellier	2	Silver	opt2	225
Montpellier	2	Platinium	opt1	250
Montpellier	2	Platinium	opt2	20
Montpellier	3	Gold	opt1	260
Montpellier	3	Gold	opt2	387.5
Montpellier	3	Silver	opt1	92.5
Montpellier	3	Silver	opt2	270
Montpellier	3	Platinium	opt1	0
Montpellier	3	Platinium	opt2	190
Montpellier	4	Gold	opt1	80
Montpellier	4	Gold	opt2	550
Montpellier	4	Silver	opt1	95
Montpellier	4	Silver	opt2	0
Montpellier	4	Platinium	opt1	275
Montpellier	4	Platinium	opt2	575
Montpellier	5	Gold	opt1	25
Montpellier	5	Gold	opt2	255
Montpellier	5	Silver	opt1	177
Montpellier	5	Silver	opt2	5
Montpellier	5	Platinium	opt1	247
Montpellier	5	Platinium	opt2	165

(d) GEO Montpellier

FIG. 1.8 – Cube de données fils de Sud sous forme tabulaire

leur sous-cube, mais qui suivent le comportement général du niveau supérieur, l'évolution est différente. Pour les distances de Manhattan et euclidiennes (médiane, min et moyenne), nous trouvons, à partir d'un certaine valeur de k, des séquences outliers qui suivent le comportement général du niveau supérieur. Ce nombre diminue ensuite dès que k tend vers le nombre de sous-cubes, c'est-à-dire quand on considère que toutes les séquences sont outliers. La mesure Cosinus identifie plus de séquences outliers à un niveau et communes au niveau supérieur.

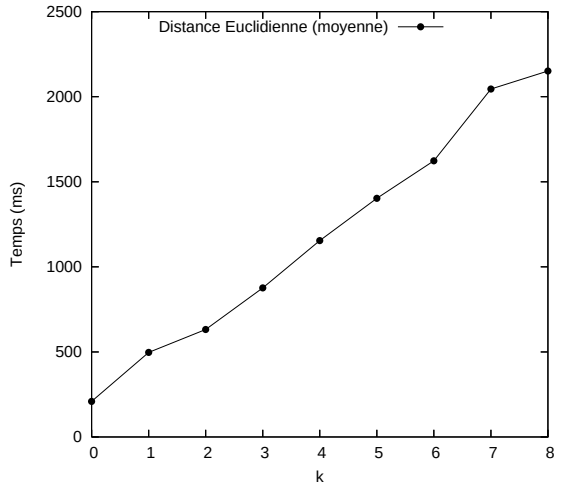
Ces premières expérimentations sont encourageantes dans la mesure où le temps d'exécution de l'algorithme est quasiment linéaire par rapport au paramètre k et deux types d'outliers sont extraits, ce qui peut faciliter et enrichir la navigation de l'utilisateur dans le cube.

1.6 Discussion

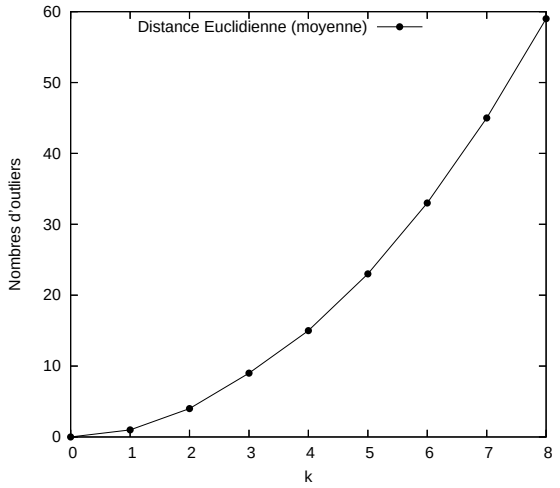
Dans ce chapitre, nous avons proposé une méthode originale d'aide à la navigation dans un cube de données. Nous avons défini des algorithmes permettant de définir les top n séquences outliers à un niveau de granularité et d'étudier plus en détails les sous-données associées à un niveau plus fin. Ainsi une séquence outlier à un niveau plus fin peut suivre ou ne pas suivre le comportement général du niveau supérieur. Des chemins de navigation sont ainsi proposés et permettent de guider l'utilisateur dans sa recherche afin qu'il cible le plus directement possible les données susceptibles de l'intéresser. Les algorithmes mis en œuvre sont suffisamment génériques pour être utilisés avec différentes mesures de distance et avec différents opérateurs d'agrégation. Dans cet chapitre, nous utilisons trois mesures (distance euclidienne, distance de Manhattan et mesure de similarité cosinus) couplées à trois opérateurs (moyenne, médiane et min). Il est évidemment possible d'utiliser d'autres mesures. Les expérimentations menées sur des cubes de données réels souligne l'intérêt de notre proposition.

Ces résultats encourageants nous incitent à approfondir ces travaux par une analyse et une utilisation plus fine de la mesure. Dans le monde réel, les cubes de données contiennent en effet souvent un nombre important de cellules vides. La prise en compte des cellules vides est une véritable problématique de recherche. Dans certains cas, la cellule vide peut être considérée comme une mesure égale à 0. Toutefois, une cellule vide est rarement équivalente à zéro. En effet, il serait injuste de mettre un zéro à un étudiant non inscrit à un module, ou de considérer qu'aucune vente n'a eu lieu dans un magasin alors que le produit n'y est pas proposé.

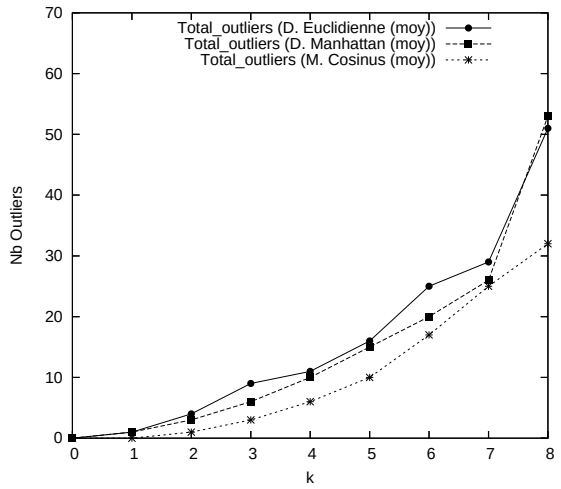
Notre approche ne permet pas d'identifier des sous-séquences comme outliers. Par exemple, si l'activité d'un centre est très atypique entre juillet de octobre, notre approche retournera la séquence contenant toute l'activité de ce centre. Il serait donc très intéressant de s'appuyer sur le découpage des séquences proposé dans [LH07] afin d'identifier des sous-séquences. A partir des types d'anomalies définis dans



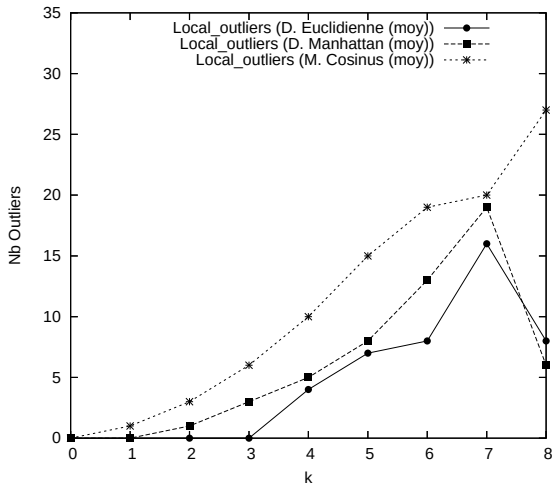
(a) Temps d'exécution en fonction du nombre de top k outliers recherchés



(b) Nombre d'outliers extraits en fonction du nombre de top k outliers recherchés.

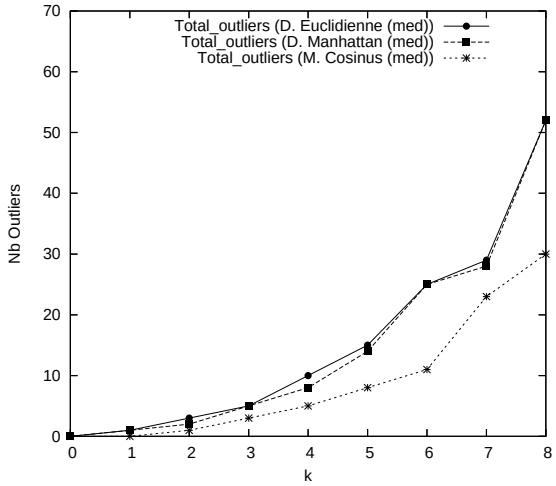


(c) Nombre d'outliers "totalement outliers" en fonction du nombre de top k outliers recherchés(moy.)

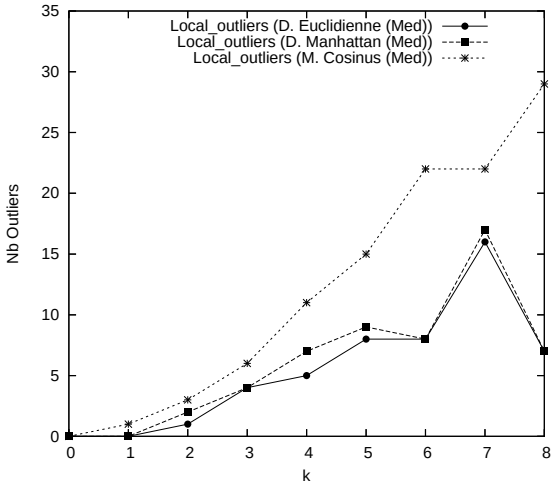


(d) Nombre d'outliers "localement" en fonction du nombre de top k outliers recherchés (moy.).

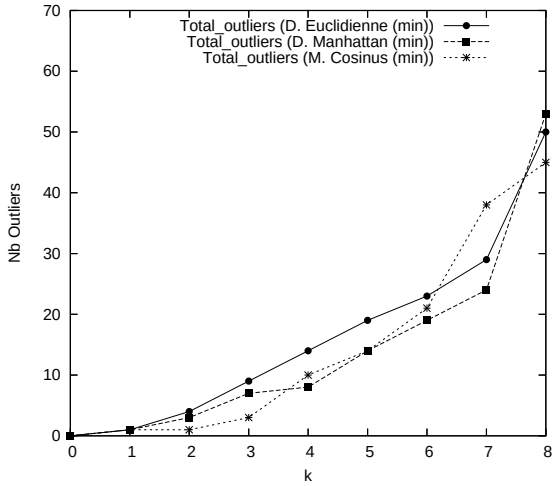
FIG. 1.9 – Expérimentations



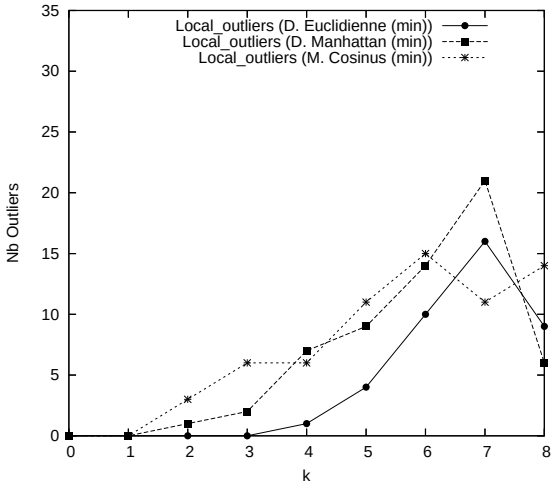
(a) Nombre d'outliers "totalement outliers" en fonction du nombre de top k outliers recherchés(med.)



(b) Nombre d'outliers "localement" en fonction du nombre de top k outliers recherchés (med.).



(c) Nombre d'outliers "totalement outliers" en fonction du nombre de top k outliers recherchés(min)



(d) Nombre d'outliers "localement" en fonction du nombre de top k outliers recherchés (min).

FIG. 1.10 – Expérimentations

[LH07], nous pouvons imaginer proposer des chemins de navigation en fonction de chaque type d'anomalie (tendance, magnitude, etc.).

Chapitre 2

Règles Inattendues

2.1 Introduction

L'extraction de motifs et de règles est un domaine très actif de la fouille de données. Ces techniques ont été appliquées dans de nombreux domaines comme l'étude du panier de la ménagère, l'analyse du comportement des internautes, la sécurité informatique, la bio-informatique et la musique. De telles applications sont rendues possibles grâce aux développement d'algorithmes efficaces. Toutefois, l'extraction de règles présente des limites non négligeables qui peuvent limiter l'utilisation de ces techniques. Le principal inconvénient provient du nombre potentiellement énorme de règles extraites qui débouche sur un second problème inhérent à la fouille de données. L'existence d'un nombre trop important de règles rend celles-ci inutilisables par un utilisateur humain dans un processus d'aide à la décision. Cet inconvénient découle directement du type de connaissance que les approches tentent d'extraire : des règles fréquentes avec une confiance élevée. Ce type de règles est très utile quand on souhaite découvrir des relations non-observées mais n'est plus adapté lorsqu'on souhaite extraire des *relations inattendues*.

De nombreux travaux affirment que l'occurrence d'un événement fréquent apporte moins d'information que l'occurrence d'un événement rare ou caché [CDF+01, YWY04, SZ05b]. Il est donc plus intéressant d'extraire des événements inattendus non fréquents que des événements fréquents mais déjà connus par l'utilisateur. Des travaux permettent de découvrir des connaissances inattendues à partir de règles d'association comme les exceptions ou les règles surprenantes [Suz99, BCM04]. Par exemple, la règle *"ceinture de sécurité et enfant de moins de 10 ans → danger"* est une règle d'exception par rapport à la règle commune *"La ceinture de sécurité sauve des vies"*.

Les données sont souvent stockées au cours du temps. Par exemple, les ventes sont rapportées tous les jours ou toutes les heures. De plus, les données sont souvent agrégées par rapport à plusieurs dimensions dans un cube de données. Par exemple, les ventes sont agrégées par rapport à l'enseigne, la ville, le type et l'âge des consommateurs, etc. Même s'il existe de nombreuses propositions autour de la découverte de règles d'association inattendues, il n'y a pas d'approche qui combine à la fois règles inattendues et temporalité dans un contexte multidimensionnel.

Dans ce chapitre, nous introduisons les *règles séquentielles multidimensionnelles inattendues*. Ce nouveau type de règle permet d'extraire des corrélations atypiques entre les événements au cours du temps dans un contexte multidimensionnel. Une règle séquentielle multidimensionnelle inattendue est non-fréquente mais possède une forte confiance. Ces règles permettent de représenter les comportements qui dévient des comportements généraux souvent connus. Ces comportements généraux sont modélisés par des règles séquentielles multidimensionnelles fréquentes qui ont une forte confiance. En effet, un comportement général correspond à des règles qui apparaissent souvent dans les données, et qui sont donc fréquentes. Ces règles doivent être également suffisamment fréquentes pour être considérées. Les règles inattendues sont *cachées* par une règle générale. Elles représentent soit des futurs comportements généraux émergeant, soit des comportements surprenants qui doivent être pris en compte.

Nous présentons notre proposition qui s'appuie sur les motifs séquentiels multidimensionnels. Nous introduisons les concepts relatifs à notre approche ainsi que les algorithmes permettant la découverte de telle connaissance. Des expérimentations menées sur des données réelles sont décrites et soulignent l'intérêt de notre proposition.

Exemple

Afin d'illustrer notre proposition, considérons l'exemple suivant qui sera utilisé dans tout ce chapitre.

Soit un cube de données DC dans lequel les transactions effectuées par des consommateurs sont agrégées. Plus précisément, DC est défini sur 5 dimensions, l'agrégation correspond aux nombres de produits vendus où :

- D correspond à la date. Les ventes sont agrégées à chaque pas de temps $(1, \ldots, 12)$.
- C représente la ville dans laquelle les ventes ont été effectuées (N.Y, L.A, etc.)
- A correspond la catégorie d'âge des consommateurs, nous considérons 3 valeurs discrètes notées Y, M et O.
- CH est le loisir principal des consommateurs (marche, surf, golf, etc.)
- P est le produit vendus (voiture, vélo, etc.).
- M représente la mesure.

Ainsi, la première cellule de DC (cf. Fig. 2.1) signifie qu'à la date 1, 12 « petites voitures » ont été achetées à $N.Y$ par des *jeunes* consommateurs dont le loisir préféré est le *golf*.

Nous souhaitons extraire toutes les règles séquentielles multidimensionnelles inattendues qui établissent des corrélations entre la catégorie d'âge, le loisirs des consommateurs et les produits achetés. Ces règles doivent respecter un seuil de confiance et de support afin de les différencier du *bruit* présent dans les données. Le calcul du support et de la confiance s'effectue à l'aide des villes où les transactions ont été effectuées. Par rapport à la partition de l'ensemble de dimensions $\mathcal{D}$ définie précédemment (cf. Def. 1.1, page 42), la partition de $\mathcal{D}$ est la suivante :

- $D_A = \{CH, P, A\}$

D Date	C City	A Age	CH Customer −Hobby	P Product	M
1	NY	Young	Golf	Little_car Bike Van	12 14 2
...	...	...	...	...	...
...	...	...	...	...	
12	L.A	Old	Walking	Bike Shoes ...	19 25 ...

FIG. 2.1 – Cube de données « exemple » DC

- $D_R = \{ C\}$
- $D_T = \{D\}$
- $D_I = \{\}$

Une règle inattendue étant une règle qui diffère d'un règle plus générale, nous devons considérer les deux types de règles : inattendues et communes.

Par exemple, une règle générale ou commune par rapport à DC peut être si *jeune consommateur a récemment reçu son permis de conduire, alors il va acheter une petite voiture*. Cette règle représente une tendance générale, elle a donc un support et une confiance élevés. Par rapport à cette règle, une règle inattendue peut être : *Si un jeune consommateur dont le loisir préféré est le surf a récemment reçu son permis de conduire, alors il va acheter un van.* Cette règle a un support faible mais une confiance forte. Par rapport à la règle commune, de l'information a été ajoutée dans le prémisse, ce qui engendre une conclusion différente et inattendue.

2.2 L'extraction de règles inattendues dans la littérature

Dans la littérature, de nombreux travaux montrent qu'un événement fréquent apporte moins d'information qu'un événement rare ou caché [CDF+01, YWY04, SZ05b]. De nombreuses approches ont donc été proposées pour découvrir des événements non fréquents mais inattendus. Le principal problème est de déterminer ce qu'est un événement intéressant et comment extraire de tels événements.

Parmi les travaux traitant de la découverte de règles intéressantes, il y a deux types d'approches. D'un côté, les *approches orientées utilisateur* requièrent l'intervention d'un expert. De l'autre côté, les *approches orientée données* essaient d'extraire de manière autonome un ensemble restreint de règles. On

utilise également les termes *subjectives* et *objectives* pour caractériser les approches orientées utilisateur et les approches orientées données [ST96].

Dans les approches orientées utilisateur, un expert doit intervenir sur au moins un des points suivants :

- Déterminer des restrictions sur les attributs qui sont susceptibles d'apparaître dans les relations [SVA97].
- Définir des hiérarchies sur les données [HF95].
- Indiquer l'utilité potentielle d'une règle par rapport à une base de croyances [LHCM00].
- Eliminer toutes les règles non pertinentes dans une première étape afin que les autres règles puissent apparaître dans les étapes suivantes [Sah99].

Les approches orientées utilisateurs sont assez contraignantes dans la mesure où elles requièrent l'intervention d'un expert dès que les données changent. C'est la raison pour laquelle nous nous focalisons sur les approches orientées données.

Les approches objectives sont également divisées en deux sous-catégories. Des travaux utilisent des mesures d'intérêt autres que le support et la confiance [Kod01, TKS04, GH06]. D'autres approches essaient de découvrir des connaissances inattendues qui ne peuvent pas être extraites par des algorithmes classiques.

Les règles porteuses d'information ne sont pas nécessairement celles qui sont fréquentes. [CDF+01] et [MH01] proposent de découvrir des itemsets non fréquents mais qui sont très fortement corrélés. Dans [ZYO03], les auteurs proposent d'extraire des « particularités » (*peculiarities* an anglais) qui sont définies comme des règles ayant une très forte confiance mais qui ne sont pas fréquentes par rapport à une mesure de voisinage. Ces particularités sont significativement différentes du reste des individus. L'approche définie dans [YWY04] permet d'extraire des séquences inhabituelles où les items ayant des faibles probabilités d'apparition apparaissent ensembles. Les séquences ainsi découvertes sont surprenantes. Cette approche n'utilise pas le support pour déterminer la fréquence de la séquence, les auteurs se basent sur des mesures d'entropies pour détecter les séquences surprenantes.

L'approche définie par Suzuki est très intéressante. Elle consiste à chercher les exceptions qui apparaissent dans une base de données [Suz99, HLSL00, Suz02a, Suz02b, SZ05b]. La présence d'un attribut interagissant avec un autre peut changer la conséquence d'une règle commune. Par exemple, la règle *"ceinture de sécurité et enfant de moins de dix ans* $\rightarrow$ *danger"* est une règle exceptionnelle par rapport à la règle commune *la ceinture de sécurité sauve des vies*. La forme générale d'une règle exceptionnelle est :

$$X \Rightarrow Y, XZ \Rightarrow \neg Y, X \not\Rightarrow Z$$

$X \Rightarrow Y$ est une *règle commune*, , $XZ \Rightarrow \neg Y$ est une règle d'exception où $\neg Y$ peut être une valeur concrète E. $X \not\Rightarrow Z$ est la règle de *référence*. Dans [HLSL00] et [SZ05b], les auteurs utilisent cinq différents paramètre pour décrire ces règles. Ils essaient de découvrir des interactions entre des attributs :

dans [Suz02a], X représente les *antibiotiques*, Y la *guérison*, Z un *staphylocoque* et E *la mort*. La règle suivante peut être ainsi découverte : *habituellement, avec l'aide d'antibiotiques, les patients guérissent mais si un staphylocoque apparaît, combiné aux antibiotiques, cela résulte à la mort du patient.* Ce type de règle est très intéressant et ne peut pas être détecté par des algorithmes classiques d'extraction de règles d'association. Les règles d'exceptions sont cachées par des règles plus générales. Il y a plusieurs méthodes pour extraire de telles règles. Certaines sont basées sur la J mesure qui correspond à la quantité $J(x;y)$ d'information compressée par une règle $y \rightarrow x$ [SG92]. Dans [SS96], les auteurs définissent une mesure d'intérêt d'un couple (règle commune, exception) comme le produit des J mesures de l'exception et de la règle commune associée. Ils souhaitent extraire les paires qui les valeurs les plus importantes. Dans [Suz02b], les auteurs définissent 5 seuils afin d'extraire ces paires. Cependant, une spécification trop stricte de ces différents seuils peut mener à la découverte d'aucune règle. Dans [Suz99], les auteurs proposent une solution à ce problème en utilisant des arbres auto-équilibrés (AVL) pour chaque seuil pour mettre à jour les valeurs de ces seuils.

Dans [BCM04], les auteurs définissent la notion de règles anormales qui sont des règles d'association qui sont vérifiées lorsque les règles générales ne le sont pas. Plus formellement, soient trois itemsets X, Y et A, $X \rightsquigarrow A$ est une règle anormale par rapport à la règle $X \Rightarrow Y$ (A est l'anomalie) si les conditions suivantes sont vérifiées :

- $X \Rightarrow Y$ est une règle forte (support et confiance élevés).
- $X\neg Y \Rightarrow A$ est une règle ayant une forte confiance.
- $XY \Rightarrow \neg A$ a également une forte confiance.

Cette approche s'appuie sur des seuils de support ($MinSupp$) et de confiance ($MinConf$) puisque les règles sont des cas particuliers de règles d'association. Sémantiquement, ce type de règle essaie de capturer les comportements suivants : *Quand X apparaît alors nous avons soit Y (habituellement) soit A (exceptionnellement).* Les règles d'anomalie représentent ainsi des déviations homogènes du comportement habituels. Par rapport à la définition de Suzuki, ces règles sont sémantiquement différente. De plus, cet approche ne nécessite pas l'existence d'un itemset « conflictuel » Z. Les algorithmes permettant la découverte de telles règles sont basées sur le paradigme *a priori*.

Il y a beaucoup d'approches dites objectives qui essaient de découvrir des types particulier de règles. Chaque type a une sémantique particulière. Aucune de ces approches ne prend en compte une relation d'ordre (le temps).

2.3 Règles Séquentielles Multidimensionnelles Inattendues

2.3.1 Présentation de la proposition

Notre but est d'extraire des règles inattendues qui sont souvent cachées par des règles communes qui ont un support important. L'idée principale, dans ce chapitre, est d'exploiter les valeurs jokers dans le

prémisse d'une règle commune. Nous souhaitons instancier au moins une valeur joker dans l'antécédent d'une règle afin de détecter une conclusion différente et inattendue.

Par exemple, la règle CR est une règle commune (fréquence et support élevés) de la forme

$$CR = P \rightarrow Q$$

P et Q sont deux séquences multidimensionnelles.

Par rapport à la règle commune CR, une règle inattendue UR est une règle UR qui n'est pas fréquente mais qui a une forte confiance :

$$UR = P_{specialized} \rightarrow Q'$$

$P_{specialized}$ est une *instanciation* de P. Au moins une valeur joker sur une dimension d_i d'un item de P a été instanciée avec une valeur du domaine de D_i. Les séquences multidimensionnelles Q et Q' sont différentes. UR n'est pas fréquente mais a une forte confiance.

Avant d'introduire la définition formelle des règles séquentielles multidimensionnelles inattendues, il est nécessaire de définir la notion de règle séquentielle multidimensionnelle, l'instanciation des valeurs joker dans le prémisse de la règle commune, et la différence de conclusions.

2.3.2 Règle séquentielle multidimensionnelle

Définition 2.1. Soit la séquence multidimensionnelle $\alpha = \langle i_1, i_2, \dots i_k, i_{k+1}, \dots, i_n \rangle$ où chaque i_j représente un itemset multidimensionnel, une *règle séquentielle multidimensionnelle* R est une implication de la forme :

$$R : \langle i_1, i_2, \dots i_k \rangle \rightarrow \langle i_{k+1}, \dots, i_n \rangle$$

Cette implication est *temporelle*, cela signifie qu'un séquence implique une autre séquence tel qu'il existe au moins un pas de temps entre la fin du prémisse et le début de la conclusion. En d'autres mots, l'implication ne pas pas couper un itemset en deux.

Comme pour les règles d'association, la pertinence d'une règle d'une règle séquentielle multidimensionnelle est indiquée par son *support* et sa *confiance*.

Définition 2.2 (Support d'une règle séquentielle multidimensionnelle)**.** Soit une règle séquentielle multidimensionnelle $R = \langle i_1, i_2, \dots i_k \rangle \rightarrow \langle i_{k+1}, \dots, i_n \rangle$, le support de R est égal à :

$$support(R) = support(\langle i_1, i_2, \dots i_k, i_{k+1}, \dots, i_n \rangle)$$

Définition 2.3 (Confiance d'une règle séquentielle multidimensionnelle)**.** Soit une règle séquentielle multidimensionnelle $R = \langle i_1, i_2, \dots i_k \rangle \rightarrow \langle i_{k+1}, \dots, i_n \rangle$, la confiance de R est égal à :

$$Conf(R) = \frac{support(\langle i_1, i_2, \dots i_k, i_{k+1}, \dots, i_n \rangle)}{support(\langle i_1, i_2, \dots i_k \rangle)}$$

2.3.3 Spécification des prémisses étoilés

Afin de découvrir des règles séquentielles multidimensionnelles inattendues, nous devons instancier des valeurs jokers (*) dans le prémisse des règles communes.

Nous définissons les fonctions suivantes :

- $*_\lambda(x)$ t.q. $*_\lambda(x) = x$
- a_{i_λ} t.q.

$$a_{i_\lambda}(x) = \begin{cases} a_i & \text{si} \quad x = a_i \\ \emptyset & \text{sinon} \end{cases}$$

Ces fonctions sont associées avec les valeurs d_i d'un item multidimensionnel $C = (d_1, d_2, \ldots, d_m)$. Nous pouvons ainsi construire la fonction C_λ :

$$X = (x_1, \ldots, x_m) \mapsto C_\lambda(X) = (d_{1_\lambda}(x_1), d_{2_\lambda}(x_2), \ldots, d_{m_\lambda}(x_m))$$

Par exemple, pour l'item $C = (a, *, c)$ nous pouvons construire la fonction associée $C_\lambda = (a_\lambda, *_\lambda, c_\lambda)$.

Définition 2.4 (Instance). Soient deux items multidimensionnels C et X, X est une *instance* de C si $C_\lambda(X) = X$.

En d'autres mots, X est une instance de C si la fonction $C_\lambda(X)$ retourne l'identité pour X.

Par exemple, $X = (a, b, c)$ est une instance de $C = (a, *, c)$ puisque $C_\lambda(X) = (a_\lambda(a), *_\lambda(b), c_\lambda(c)) = (a, b, c) = X$. Nous pouvons également remarquer que X est une instance de lui même ($X_\lambda(X) = X$).

Définition 2.5 (Pseudo-instance d'un itemset).
Soient deux itemsets multidimensionnels $i = \{e_1, e_2, \ldots, e_m\}$ et $i' = \{e'_1, e'_2, \ldots, e'_{m'}\}$ tels que $m \leq m'$, i est une *pseudo-instance* de i' s'il existe m entiers $1 \leq k_1 \leq k_2 \leq \ldots \leq k_m \leq m'$ tels que $\forall e_j \in i$, $e'_{k_{j\lambda}}(e_j) = e_j$.

Un itemset i est une pseudo-instance d'un itemset i', si chaque item de i est une instance d'un item de i'. Nous utilisons le terme *pseudo* car i peut être plus petit que i'. Si, i et i' ont la même taille, alors on parle d'instance. Par exemple, $\{(a, b, c)\}$ est une pseudo-instance de $\{(a, *, c), (*, b, b)\}$ alors que $\{(a, b, c), (*, b, b)\}$ est une instance de $\{(a, *, c), (*, b, b)\}$ puisqu'ils ont la même taille.

Définition 2.6 (Pseudo-instance d'une séquence).
Soient deux séquences multidimensionnelles $s = \langle i_1, i_2, \ldots i_m \rangle$ et $s' = \langle i'_1, i'_2, \ldots i'_{m'} \rangle$ telles que $m \leq m'$, s est une *pseudo-instance* de s' s'il existe des entiers $1 \leq k_1 \leq k_2 \leq \ldots \leq k_m \leq m'$ tels que $\forall i_j \in s$, i_j est une pseudo instance de i'_{k_j}.

Par exemple, la séquence $s = \langle \{(a, b, c)(a, *, d)\}\{(a, b, *)(b, *, *)\} \rangle$ est une pseudo-instance de la séquence $s' = \langle \{(a, b, c)(a, *, d), (d, e, f)\}\{(a, *, *)(b, *, *)\} \rangle$.

Afin de découvrir des règles inattendues, nous souhaitons découvrir des règles séquentielles multidimensionnelles où quand on ajoute de l'information dans l'antécédent, la conséquence est modifiée et devient peut être inattendue. Nous devons donc définir une opération d'instanciation dans le but d'ajouter de l'information dans le prémisse d'une règle séquentielle multidimensionnelle. Cet ajout d'information se concrétise par la substitution de valeurs jokers des items de la séquence antécédent par des valeurs appartenant aux domaines des dimensions concernées. Cette opération est une spécification d'au moins une valeur joker d'un item d'une séquence s' par rapport à son instance de la séquence s (s' est une pseudo-instance de s).

Définition 2.7 (Instanciation).
Soient deux séquences multidimensionnelles s et s' telles que s' est une pseudo-instance de s, la fonction $\iota(s', s)$ est la substitution d'au moins un item e_i de s' par une instance de e_i de s.

$$\iota : séquence \times séquence \rightarrow ensemble\ de\ séquences$$
$$\iota(s', s) \mapsto \{s'' \text{ t.q. les conditions suivantes sont vérifiées } \}$$

- s'' est une instance de s'.
- $\exists\ items\ e''_i \in s'', e'_i \in s'\ et\ e_i \in s\ t.q.\ e_i$ est une instance de e'_i et $e''_i = [e_i/e'_i]$ où $[e_i/e'_i]$ est la substitution de e'_i par e_i.

Par exemple, $\iota(\langle\{(a, *, c), (e, f, *)\}\{(*, c, d)\}\rangle, \langle\{(a, b, c)\}\rangle)$ est égal au singleton $\{\langle\{(a, b, c), (e, f, *)\}\{(*, c, d)\}\rangle\}$. La valeur joker de l'item $(a, *, c)$ a été instanciée par la valeur b en fonction de son instance (a, b, c).

L'opération d'instanciation ne retourne pas une unique séquence mais un ensemble de séquences. Cet ensemble peut rendre sa gestion difficile. Toutefois, en pratique, nous instancierons de manière gloutonne, item par item afin d'obtenir à chaque fois un singleton.

2.3.4 Différence dans la conséquence

Dans le but de découvrir des règles séquentielles multidimensionnelles inattendues, nous devons découvrir une conclusion différente à partir d'un prémisse instancié. La conséquence d'une règle séquentielle multidimensionnelle est également une séquence, nous devons formaliser la différence entre deux séquences.

Définition 2.8 (Différence).
Soient deux séquences multidimensionnelles $s = \langle i_1, i_2, \ldots, i_l\rangle$ et $s' = \langle i'_1, i'_2, \ldots, i'_{l'}\rangle$ sont dites *différentes* $(s \neq s')$ si $s \not\preceq s'$ et $s' \not\preceq s$.

Si nous considérons que deux séquences comparables sont différentes, nous risquons de découvrir des règles où la conclusion de la règle inattendue est simplement une spécialisation de la conséquence de la règle commune. Or une spécialisation peut facilement s'expliquer par le fait que le support de la règle

« inattendue » est faible. Le caractère « inattendu » n'est pas justifié dans ce cas. Nous considérons donc que deux séquences sont différentes si elles ont au moins un item différent, et si elles sont incomparables. Par exemple, les séquences $s_1 = \langle\{(a_1, b_1)\}\{(a_2, *)\}\rangle$ et $s_2 = \langle\{(a_1, b_1)\}\rangle$ ne sont pas considérées comme différentes puisque $s_2 \preceq s_1$. Les séquences s_1 et $s_3 = \langle\{(a_2, b_2)\}\{(a_1, b_2)(a_2, b_1)\}\rangle$ sont différentes et non comparables puisque $s_1 \npreceq s_3$ et $s_3 \npreceq s_1$.

2.3.5 Règles séquentielles multidimensionnelles inattendues

Nous avons défini la spécification d'une séquence pour ajouter de l'information dans l'antécédent d'une règle commune afin de découvrir une conclusion différente et inattendue. La différence est définie dans la définition Def. 2.8 alors que la caractère inattendu sera détecté à l'aide de seuils de support.

Nous considérons cinq seuils définis *a priori* :

- $minCR$: le seuil de support minimum des règles séquentielles multidimensionnelles communes,
- $maxUR$: le seuil de support maximum des règles séquentielles multidimensionnelles inattendues,
- $minUR$: le seuil de support minimum des règles séquentielles multidimensionnelles inattendues,
- $minConf$: le seuil de confiance minimum des règles séquentielles multidimensionnelles (communes ou inattendues).

Le seuil $minCR$ représente la valeur de support minimale au dessus de laquelle une règle est considérée comme *fréquente*.

Le seuil $minConf$ spécifie la confiance minimale d'une implication afin que celle-ci soit considérée comme règle séquentielle multidimensionnelle. Ce seuil s'applique pour n'importe quel type de règle (commune ou inattendue) car une règle est définie avant tout comme une implication forte.

Nous considérons qu'une règle qui ne respecte pas le seuil $minCR$ n'est pas nécessairement une règle inattendue potentielle. Nous définissons ainsi deux seuils de support supplémentaires. Si le support d'une règle est supérieur à $maxUR$, alors la règle est *trop fréquente* pour pouvoir être une règle inattendue. En effet, rappelons que la caractère *inattendu* d'une règle est lié à la non fréquence de celle-ci. Nous proposons ce seuil afin de souligner cette différence. Toutefois, ces seuils étant définis *a priori*, $maxUR$ peut être égal à $minCR$. De façon similaire, nous utilisons le seuil de support $minUR$ pour différencier l'inattendu du *bruit*.

La figure 2.2 illustre l'utilisation de ces différents seuils de support. Nous pensons qu'une règle inattendue doit suffisamment apparaître dans les données pour ne pas être considéré comme du bruit. De plus, le support d'une règle inattendue doit être suffisamment faible pour ne pas être que celle-ci ne soit pas considérée comme une règle commune.

En fonction de ces différents seuils, nous pouvons définir formellement le problème de l'extraction de règles séquentielles multidimensionnelles inattendues.

Un règle séquentielle multidimensionnelle dite *commune* CR est une règle ayant un support et une confiance élevés :

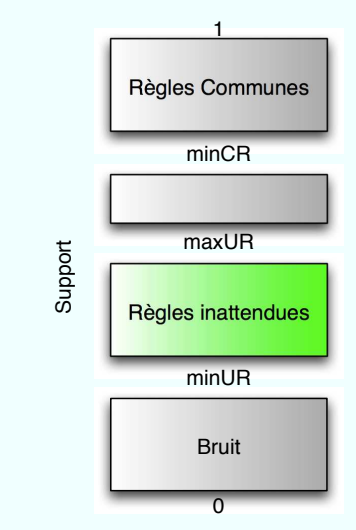


FIG. 2.2 – Illustration des différents seuils de support

$$CR = s_\alpha \rightarrow s_\beta \; t.q. \; support(CR) > minCR \; et \; confiance(CR) > minConf \tag{2.1}$$

Soit une instance fréquente de l'antécédent de CR :

$$s' \; t.q \; support(s') > minCR \; et \; s' \; est \; une \; instance \; de \; s_\alpha \tag{2.2}$$

Une règle séquentielle multidimensionnelle inattendue est une règle ayant une forte confiance et dont le prémisse est une instanciation de s_α, vérifiant les conditions suivantes :

$$UR = \iota(s_\alpha, s') \rightarrow s_c \, t.q. \, confiance(UR) > minConf, \; minUR \leq support(UR) \leq maxUR \, et \, s_c \neq s_\beta \tag{2.3}$$

Il faut vérifier qu'il n'y pas de règle commune ayant comme antécédent s_α et comme conséquence s_c. S'il existe une telle règle, UR ne peut pas être considérée comme inattendue.

$$VR = s_\alpha \rightarrow s_c \; t.q. \; confiance(VR) < minConf \; et/ou \; support(VR) < maxUR \tag{2.4}$$

Pour découvrir l'ensemble complet des règles séquentielles multidimensionnelles inattendues, il faut, pour chaque règle commune RC (2.1), découvrir l'ensemble de séquences S vérifiant (2.2), (2.3) et (2.4).

2.4 Algorithme

Dans cette partie, nous définissons un algorithme permettant l'extraction des règles séquentielles multidimensionnelles inattendues.

L'algorithme 18 décrit la mise en œuvre de notre proposition. Cet algorithme est divisé en trois étapes. Tout d'abord, les séquences multidimensionnelles sont extraites et stockées dans un arbres des préfixes. Ensuite, les règles communes sont découvertes dans l'arbre des préfixes. Finalement, la dernière étape vise à rechercher les règles inattendues.

Les séquences multidimensionnelles qui vérifient la contrainte de support $minUR$ sont extraites. Ces séquences sont stockées dans un arbre des préfixes. La méthode $getFreqSet()$ de l'algorithme 18 permet d'effectuer cette tâche. Cette méthode est certainement l'étape la plus coûteuse de l'algorithme. En effet, elle nécessite plusieurs passage sur les données afin d'extraire l'ensemble complet des séquences qui ont un support vérifiant le seuil $minUR$. Toutefois, on peut imaginer appeler cette méthode une seule fois avec un seuil suffisamment faible pour autoriser la mise en œuvre multiple des autres étapes avec des paramètres différents. L'arbre des préfixes retourné étant la base des étapes suivantes, on peut découvrir des règles communes et des règles inattendues en fonction de différents choix des paramètre ($minUR, maxUR, minConf$ and $minCR$). On peut imaginer, par exemple, de réaliser l'extraction des séquences chaque nuit, l'utilisateur recherchant des règles inattendues durant la journée.

Les règles séquentielles multidimensionnelles communes sont extraites sans aucun passage supplémentaire sur les données. En effet, la méthode $getCR()$ considère chaque nœud de l'arbre des préfixes $freqTree$ une seule fois (un seul parcours de l'arbre). Cette méthode retourne l'ensemble complet des règles séquentielles multidimensionnelles qui respectent les seuils de confiance $minConf$ et de support $minCR$.

De façon similaire, $freqTree$ est parcouru afin de découvrir l'ensemble des règles séquentielles multidimensionnelles inattendues. Toutefois, la découverte de règle inattendue doit aussi considérer l'ensemble des règles communes (noté CRS). En effet, $freqTree$ est parcouru afin d'extraire des règles $r : p \rightarrow q$ dont le support et la confiance vérifient les seuils $minUR$, $maxUR$ et $minConf$ ($minUR \leq support(r) \leq maxUR$ et $confiance(r) \geq minConf$). De plus, pour être considérée comme une règle inattendue potentielle, il est nécessaire de trouver un prémisse p' dans l'ensemble des règles communes CRS tel que p est une instance de p'. La règle r est alors potentiellement inattendue. Pour déterminer si elle est réellement inattendue, il faut vérifier qu'il n'existe pas de règle commune $p' \rightarrow q$ dans CRS. Si cette condition est vérifiée, la règle r est ajoutée à l'ensemble des règles séquentielles multidimensionnelles inattendues URS. Sinon, r n'est pas une règle inattendue, car une règle commune (fréquente) ayant comme antécédent p' a la même conséquence. Les différentes opérations effectuées sur l'ensemble CRS peuvent être améliorées en utilisant des structures plus élaborées comme des tables de hachage avec des hachages simples ou doubles.

Algorithme 18 : Mining Unexpected Multidimensional Sequential Rules

Entrées : $DC, D_A, D_T, D_R, D_F, minCR, maxUR, minUR, minConf$

Sorties : L'ensemble URS des règles séquentielles multidimensionnelles inattendues

début

$FreqTree \leftarrow getFreqSeq(DC, D_A, D_R, D_T, minUR)$

$CRS \leftarrow getCR(FreqTree, minCR, minConf)$

$URS \leftarrow \emptyset$

pour chaque *rule* $r : p \rightarrow q \in freqTree$ *s.t.* $minUR \leq supp(r) \leq maxUR$ *and* $conf(r) \leq minConf$ **faire**

  si $\exists$ *premise* $p' \in CRS$ *s.t.* p *is an instance of* p' **alors**

    si $\exists seq\ x$ *s.t.* $\iota(p', x) \rightarrow p$ *and* $supp(x) \geq minCR$ **alors**

      si $\nexists p' \rightarrow q \in CRS$ **alors**

        $URS \leftarrow URS \cup \{r\}$

fin

2.5 Expérimentations

Dans cette section, nous rapportons des expérimentations effectuées sur des données réelles. Ces expérimentations ont pour objectif de montrer la pertinence de cette proposition. Les expérimentations ont été effectuées sur des données marketing issues d'EDF décrites l'annexe A.

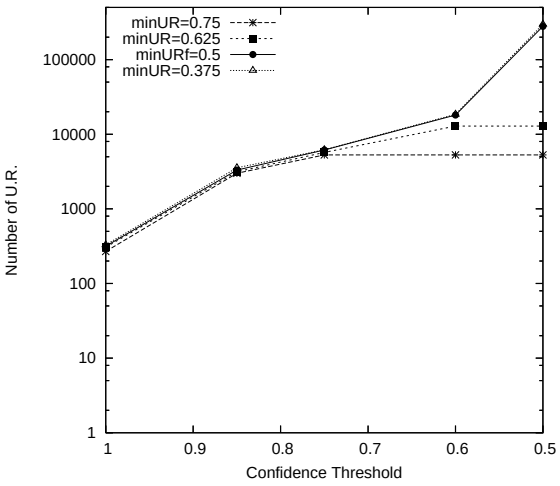
Le cube de données contient six dimensions d'analyse. Notons que la mesure est partitionnée et considérée comme dimension d'analyse.

Par rapport à ces données, une règle commune est :
$CR = \langle\{(*, FLUX, *, *, *)\}\{(*, *, *, Electrical, *)\}\rangle \rightarrow \langle\{(*, FLUX, *, *, *)\}\{(BIEN, *, *, *, *\}\rangle$.

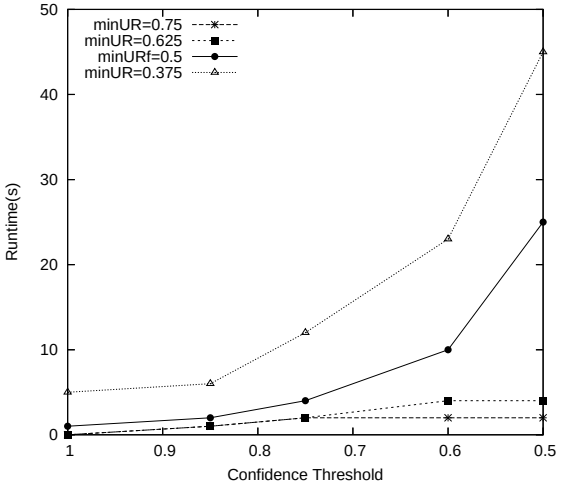
Par rapport à CR, une règle séquentielle multidimensionnelle inattendue est
$UR = \langle\{(\textbf{BIEN}, FLUX, *, *, *)\}\{(*, *, *, Electrical, *)\}\rangle \rightarrow \langle\{(*, *, Phone, Electrical, *)\}\rangle$.

Dans le but de montrer l'intérêt de notre approche, nous rapportons le comportement de notre proposition (temps d'exécution nombre de règles inattendues, le ratio entre le nombre de règles inattendues et le nombre total de règles) en fonction de plusieurs seuils ($minUR$, $minConf$).

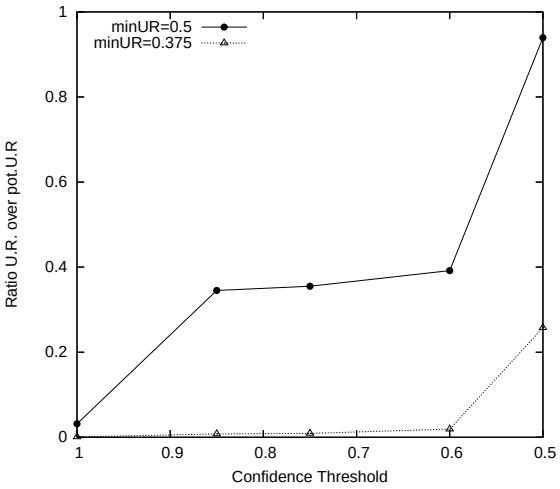
La Fig. 2.3(b) montre le temps d'exécution de notre proposition pour extraire des règles séquentielles multidimensionnelles inattendues dans le cube de données pour quatre seuils $minUR$ différentes en fonction du seuil de confiance minimum. Le comportement des courbes est assez similaire à celui observé sur la Fig. 2.3(e) où le temps d'exécution est rapportée en fonction de $minUR$ pour quatre seuils de confiance différents. En effet, quand le support devient faible, le nombre de règles inattendues (Figure 2.3(a) et 2.3(d)) ou potentiellement inattendues augmente. Plus faibles sont les seuils, plus grand est l'espace de recherche.



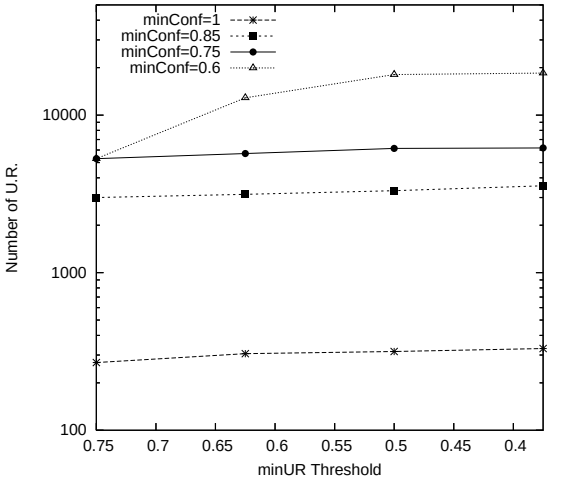
(a) Nombre de règles inattendues par rapport au seuil de confiance ($maxUR = 0.8$)



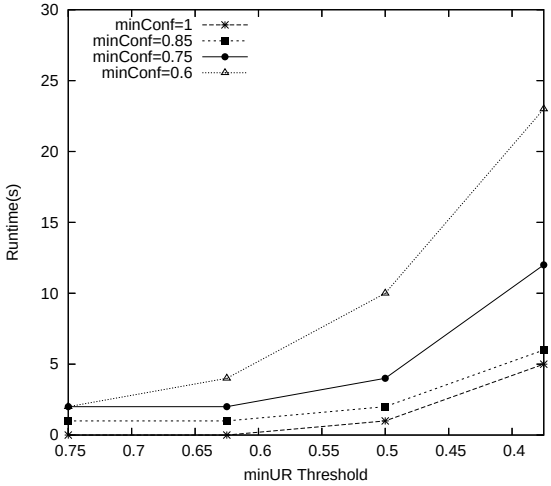
(b) Temps d'exécution par rapport au seuil de confiance ($maxUR = 0.8$)



(c) Pourcentage de règles inattendues par rapport au seuil de confiance ($maxUR = 0.8$)



(d) Nombre de règles inattendues par rapport au seuil $minUR$ ($maxUR = 0.8$)



(e) Temps d'exécution par rapport au seuil $minUR$ ($maxUR = 0.8$)

FIG. 2.3 – Expérimentations menées sur des données réelles

La Fig. 2.3(c) montre le pourcentage de règles séquentielles multidimensionnelles inattendues en fonction du seuil de confiance. Par exemple, pour $minUR = 0.375$, seulement 1% des règles dont le support est compris entre $minUR$ et $maxUR$ sont inattendues. Quand le seuil de confiance devient trop faible, il ne représente pas une contrainte suffisante et le pourcentage de règles inattendues devient important. Si l'allégorie de la fouille de données est la découverte de diamants, l'extraction de règles inattendues est très proche de cette image lorsque le seuil de confiance est suffisamment contraignant.

2.6 Discussion

Dans ce chapitre, nous définissons les règles séquentielles multidimensionnelles inattendues. Les règles séquentielles multidimensionnelles inattendues combinent à la fois le temps (séquence), la multidimensionnalité et l'atypicité. Une règle inattendue est une règle non fréquente qui est cachée par une règle commune (fréquente). Les règles communes modélisent les comportements généraux alors que les règles inattendues caractérisent les comportements qui ne suivent pas les comportements généraux. Les règles inattendues peuvent donc identifier soit des nouvelles tendances qui ne sont pas encore suffisamment fréquentes pour être considérer comme des règles communes, soit des comportement qui dévient des comportements classiques. Dans les deux cas, il est très intéressant des les desceller afin d'anticiper les tendances, ou de cibler les comportements atypiques afin de proposer des solutions adaptées. Nous définissons ainsi les différentes concepts associées (règles séquentielles multidimensionnelles, spécification des prémisses, etc.) ainsi qu'un algorithme permettant l'extraction de telles règles. Des expérimentations menées sur des données réelles sont rapportées et soulignent l'intérêt de cette proposition.

Cette proposition offre de nombreuses perspectives. Les expérimentations ont été effectuées avec la mesure discrétisée en dimension d'analyse. Nous pouvons effectuer des expérimentations en utilisant la mesure pour compter le support des séquences multidimensionnelles comme nous l'avons vu dans le chapitre 3 de la partie II. Nous pouvons également prendre en compte les hiérarchies dans la spécification des antécédents des règles communes. Il serait intéressant de pouvoir extraire les règles les plus inattendues (top k). Enfin, nous souhaitons proposer une extension de l'approche de Suzuki qui permet d'extraire des exceptions dans un contexte de règles d'association. Pour cela, il faut établir une union efficace entre deux séquences multidimensionnelles.

Bilan et Perspectives

Dans cette partie, nous montrons la nécessité de pouvoir extraire des comportements atypiques. Les comportements atypiques peuvent être le résultat d'une action frauduleuse (attaque informatique, fraude bancaire, etc.) ou peuvent identifier des comportements d'individus qui ne s'inscrivent pas dans les modèles généraux et qu'il est nécessaire de prendre en compte (clients mécontents, clients potentiels, etc.). La détection de ces comportements est également utile en médecine. Les comportements atypiques peuvent être interprétés de deux manières différentes.

1. Un comportement atypique peut être vu comme un élément ou un ensemble d'éléments qui se démarquent sensiblement des autre objets de l'ensemble de données. Dans ce cas, de tels éléments sont des outliers et représentent des données brutes.
2. Un comportement atypique peut être vu comme un comportement qui dévie du comportement général. Les comportements généraux sont modéliser par des règles ayant un fort support et une forte confiance. Dans ce contexte, les comportements atypiques sont des règles qui contredisent ces règles générales ou communes.

Dans cette partie, nous nous sommes intéressés à ces deux interprétations en prenant en compte plusieurs spécificités telles que la multidimensionnalité et la présence d'une dimension temporelle.

Dans le chapitre 1, nous proposons une méthode de navigation dans un cube de données guidée par la recherche de séquences outliers à différents niveaux de granularité. Nous employons les termes de séquences totalement outliers pour les séquences outliers à un niveau de hiérarchie L et également outlier au niveau plus général $L-1$. Dans le cas contraire, si une séquence outlier à un niveau L n'est pas outlier au niveau $L-1$ (comportement normal), alors nous parlons de séquence localement outlier. Des chemins de navigation sont ainsi proposés et permettent de guider l'utilisateur dans sa recherche afin qu'il cible le plus directement possible les données susceptibles de l'intéresser. Les algorithmes mis en œuvre sont suffisamment génériques pour être utilisés avec différentes mesures de distance et avec différents opérateurs d'agrégation. Par exemple, nous utilisons trois mesures (distance euclidienne, distance de Manhattan et mesure de similarité cosinus) couplées à trois opérateurs (moyenne, médiane et min). Des expérimentations menées sur des données réelles montrent l'intérêt de notre proposition.

Les perspectives associées à cette proposition son nombreuses. Il est nécessaire de s'intéresser au problèmes des cellules vides. Il est également primordial d'améliorer la qualité des outliers identifiés. Nous

pouvons ainsi nous intéresser à la découverte de sous-séquences outliers et utiliser d'autres mesures de distance permettant d'identifier plus finement des anomalies (tendance, magnitude, etc.) [LH07].

Dans le chapitre 2, nous définissons les règles séquentielles multidimensionnelles inattendues. Ces règles combinent à la fois le temps (séquence), la multidimensionnalité et l'atypicité. Une règle inattendue est une règle non fréquente qui est « cachée » par une règle générale (fort support et forte confiance). Les règles générales permettent de modéliser les comportements généraux alors que les règles inattendues identifient les comportements déviant des comportements généraux. Les règles inattendues peuvent donc soit identifier de nouvelles tendances qui ne sont pas encore suffisamment fréquentes pour être considérées comme des règles communes, soit des comportements atypiques qui dévient des comportements généraux. Dans les deux cas, il est primordial pour un utilisateur de les détecter afin d'anticiper les nouvelles tendances ou proposer des solutions adaptées aux comportements atypiques. Nous définissons les différents concepts associés (règles séquentielles multidimensionnelles, spécification des prémisses, etc.) ainsi qu'un algorithme permettant l'extraction de telles règles. Des expérimentations menées sur des données réelles sont décrites et montrent l'intérêt de cette proposition.

La prise en compte des hiérarchies dans spécification des antécédents des règles communes permet d'extraire des règles inattendues plus précises. Elle permet d'identifier le niveau précis de la hiérarchie où le comportement devient atypique. L'extraction des règles les plus atypiques (top k) doit être prise en compte afin d'améliorer le processus décisionnel. Une extension de l'approche de Suzuki peut s'avérer fort bénéfique dans un contexte de séquences multidimensionnelles. Il est toutefois nécessaire de définir des opérations efficaces d'union de séquences.

Ces deux propositions s'appuient sur les mêmes motivations : un événement rare doit être pris en compte. Nous pouvons imaginer extraire également des motifs séquentiels multidimensionnels, c'est-à-dire rechercher des outliers parmi les séquences multidimensionnelles fréquentes.

La détection de comportements atypiques est souvent associée à des problématiques « temps réel ». En effet, il est inutile de détecter une attaque sur un système quelques jours après l'incidents. La détection de comportements atypiques vise à empêcher les fraudes. Il faut donc les détecter à la volée afin d'offrir à l'utilisateur une plus grande réactivité. Ils serait donc très intéressant de découvrir des comportements atypiques dans des flots de données multidimensionnelles.

Nous pouvons imaginer combiner séquences outliers et règles inattendues afin de réaliser un modèle de prédictions de comportements atypiques, c'est-à-dire détecter le comportement atypique le plus tôt possible afin de limiter les répercussions négatives d'un tel comportement dans des contextes spécifiques (attaques, fraudes, etc.).

Bilan Général et Perspectives

La succession de chercheurs est comparable à un seul homme qui apprend indéfiniment
Blaise Pascal (1670) — *Pensées*

Pour conclure ce mémoire, nous proposons de faire un bilan de nos contributions et discuter des perspectives associées à cette thèse.

Conclusion

Nous avons vu dans la discussion de l'état de l'art sur les motifs qu'il n'y avait aucune proposition intégrant réellement le temps sur toutes les dimensions. Ainsi, dans les approches de l'état de l'état de l'art p. 25, une seule dimension peut varier au cours du temps.

Le constat précédent nous a conduit à proposer une nouvelle définition des motifs séquentiels multidimensionnels dans la chapitre 1 de la partie I. Cette définition permet d'extraire des relations entre des éléments de plusieurs dimensions au cours du temps. Les éléments de chaque dimension sont susceptibles de varier au cours du temps. De plus, la partition de l'ensemble de dimensions offre à l'utilisateur une plus grande liberté dans le choix des différents axes.

La nouvelle définition des motifs séquentiels multidimensionnels que nous avons introduite permet ainsi une meilleure appréhension des données analysées. Le problème d'extraction associé est une généralisation du problème d'extraction de motifs séquentiels classiques [AS95] et des motifs séquentiels multidimensionnels introduits par [PHP+01]. Toutefois, l'espace de recherche associé à cette problématique est très important du fait la multidimensionnalité et de la prise en compte du temps. Nous nous sommes attaqués au problème de deux façons différentes.

- Dans le chapitre 2, nous avons ciblé des parties de l'espace de recherche en extrayant les motifs séquentiels multidimensionnels à partir des items les plus spécifiques à l'aide de l'algorithme M^2SP. Cette méthode ne permet pas d'extraire l'ensemble complet des motifs séquentiels multidimensionnels mais se présente comme un bon compromis entre le passage à l'échelle de l'extraction et la qualité des connaissances extraites.

– Dans le chapitre 3, nous avons proposé d'extraire l'ensemble complet des motifs séquentiels multidimensionnels en nous appuyant sur une représentation condensée : les motifs clos. Nous avons défini les motifs séquentiels multidimensionnels clos qui permettent de parcourir efficacement l'espace de recherche en proposant des propriétés supplémentaires d'élagage de l'espace de recherche. L'extraction de tels motifs offre également une représentation non redondantes des connaissances extraites. Nous avons défini deux algorithmes d'extraction de motifs séquentiels multidimensionnels clos *CMSP_Cand* et *CMSP_Free* qui sont respectivement inspirés des algorithmes d'extraction de motifs séquentiels classiques *CloSpan* et *BIDE*.

Les premières contributions, formulées dans la partie I, permettent d'extraire des motifs séquentiels multidimensionnels dans des jeux de données multi-attributs ou multidimensionnels évoluant par rapport à une relation d'ordre (temps). Cependant, les données sont fréquemment décrites à l'aide des hiérarchies et des valeurs numériques sont également présentes. Dans la partie II, nous avons proposé la prise en compte de ces spécificités dans l'extraction de motifs séquentiels multidimensionnels. Les chapitres 1 et 2 s'attaquent à la gestion des hiérarchies. Nous avons défini le concept de motif séquentiel multidimensionnel h-généralisé qui permettent d'identifier des séquences définies sur différents niveaux de hiérarchies. L'algorithme M^3SP permet d'extraire de tels motifs à partir des items h-généralisés les plus spécifiques. L'algorithme *M2S_CD* offre un autre point de vue sur l'utilisation des hiérarchies en proposant d'extraire des séquences convergentes ou divergentes. Dans de telles séquences, les items se spécialisent (convergent) ou se généralisent (divergent) le long de la séquence. Dans le chapitre 3, nous avons proposé trois approches différentes de prendre en compte la présence de valeur numérique (mesure).

– Des contraintes d'agrégat peuvent être mises sur la dimension numérique afin de réduire l'espace de recherche aux données satisfaisant cette contrainte.
– La ou les dimensions numériques peuvent être discrétisées à partir de partitions strictes ou floues.
– La mesure est directement utilisée pour calculer le support d'une séquence.

Les propositions formulées dans les parties I et II permettent de découvrir des motifs séquentiels multidimensionnels dans des jeux de données multidimensionnels avec ou sans hiérarchies et/ou valeurs numériques. Ces motifs permettent d'extraire les comportements généraux et découvrir les principales tendances à partir des données examinées. De telles connaissances s'avèrent très utiles quand l'utilisateur souhaitent découvrir des relations non observées ou les tendances générales des sur les données. Toutefois, ces connaissances ne sont pas suffisantes dans certains contextes. Il faut également détecter des comportements atypiques .

Les comportements atypiques peuvent être vus comme des éléments du jeu de données examiné qui se démarquent des autres. Dans ce cas, un comportement atypique réfère une séquence de données de la base. Un comportement atypique peut également être vu comme une connaissance qui se démarque des autres connaissances. Dans cette interprétation, un comportement atypique est alors un motif ou une règle (meta séquence).

Dans le chapitre 1 de la partie III, nous avons proposé une méthode de navigation dans un cube de données guidée par la recherche de séquences outliers à différents niveaux de granularité. Nous avons employé les termes de séquences totalement outliers pour les séquences outliers à un niveau de hiérarchie L et également outlier au niveau plus général $L-1$. Dans le cas contraire, si une séquence outlier à un niveau L n'est pas outlier au niveau $L-1$ (comportement normal), alors nous avons parlé de séquence localement outlier. Des chemins de navigation ont ainsi été proposés et permettent de guider l'utilisateur dans sa recherche afin qu'il cible le plus directement possible les données susceptibles de l'intéresser. Les algorithmes mis en œuvre sont suffisamment génériques pour être utilisés avec différentes mesures de distance et avec différents opérateurs d'agrégation. Nous avons utilisé ainsi plusieurs mesures (distance euclidienne, distance de Manhattan et mesure de similarité cosinus) couplées à plusieurs opérateurs (moyenne, médiane et min).

Dans le chapitre 2 de la partie III, nous avons défini les règles séquentielles multidimensionnelles inattendues qui peuvent être vues comme un application de l'extraction de motifs séquentiels multidimensionnels. Ces règles, qui sont non fréquentes mais qui ont une forte confiance, sont « cachées » par des règles plus générales. Elles identifient des comportements différents des comportements généraux. Ces comportements atypiques peuvent identifier de futures modèles généraux ou représenter des individus « marginaux ».

Les différentes propositions de cette thèse permettent d'extraire des connaissances alliant temporalité et multidimensionnalité qui offrent une meilleure appréhension des données examinées. De nombreuses améliorations et extensions sont décrites dans les perspectives que nous détaillons dans la section suivante.

Perspectives

Les travaux effectués dans cette thèse peuvent se poursuivre de différentes façons. Nous détaillons ici quelques perspectives qui nous tiennent à cœur. Nous discutons des perspectives à court terme visant à améliorer, soit l'efficacité de l'extraction, soit la qualité des connaissances extraites. Enfin, nous distinguons les perspectives à long terme qui orientent nos travaux vers d'autres contextes ou d'autres types de connaissances.

Améliorer l'efficacité de l'extraction

Nous avons vu dans les chapitres précédents que le passage à l'échelle des algorithmes d'extraction de motifs séquentiels multidimensionnels était remis en cause lorsque le nombre de dimensions d'analyse était trop important. Il serait donc très intéressant de travailler sur la réduction du nombre de dimensions d'analyse avant d'effectuer l'extraction de motifs séquentiels multidimensionnels. On peut par exemple se baser sur la détection de dépendances entre les dimensions afin de supprimer des dimensions qui dépendent d'autres.

Nous avons également vu que l'utilisation d'une représentation condensée permettait d'élaguer plus efficacement l'espace de rechercher et d'éviter ainsi de considérer des séquences non-prometteuses. Nous avons défini des algorithmes d'extraction de motifs séquentiels multidimensionnels clos. Dans le contexte des itemsets, des travaux d'extraction de motifs clos s'appuient sur de puissants outils mathématiques relatifs à l'analyse des concepts formels (Formal Concept Analysis en anglais noté FCA). Il serait intéressant de comparer les algorithmes $CMSP$ par rapport à un algorithme basé sur FCA. L'utilisation d'autres représentations condensées peut être envisagées. Même si l'adaptation de celles basées sur le support (dérivables, k libres, etc.) est impossible dans le contexte des séquences et que les clos restent actuellement la meilleure solution, il n'existe actuellement aucune preuve théorique garantissant que les clos sont les plus adaptés pour élaguer l'espace de recherche.

Pour améliorer l'efficacité de l'extraction de motifs séquentiels multidimensionnels, nous pouvons nous appuyer sur l'extraction de motifs séquentiels par approximation. En effet, si on accepte de ne pas avoir la solution exacte, les approximations s'avèrent un excellent compromis. Ils existent des travaux sur l'extraction de séquences par approximation qui s'appuient sur l'échantillonnage du jeux de données ou sur des mesures de similarités entre séquences (distance de Levenstein, etc.). Il serait intéressant d'adapter ces approches dans un contexte multidimensionnel.

Comme l'algorithme SPIRIT [GRS99] qui permet de cibler la recherche de motifs séquentiels classiques sur ceux qui sont jugés intéressants par l'utilisateur, nous pouvons prendre en compte les préférences des utilisateurs afin de ne pas perdre de temps à l'extraction de motifs jugés inintéressants par l'utilisateur. L'intérêt de cette méthode couplée aux hiérarchies ne fait aucun doute. De surcroît, elle permettrait de coupler l'efficacité de l'extraction à la qualité des connaissances extraites.

Améliorer la qualité des connaissances extraites

Dans les chapitres 2 de la partie I et 1 de la partie II, nous proposons d'extraire des motifs séquentiels multidimensionnels à partir des items les plus spécifiques. Il serait judicieux d'extraire des séquences à partir d'autres caractéristiques. Par exemple, on peut imaginer extraire des motifs séquentiels à partir des items fréquents clos.

Nous avons également vu que la prise en compte des hiérarchies permettait d'améliorer la qualité des connaissances puisque les motifs extraits sont définis suivant plusieurs niveaux de hiérarchies. Dans notre proposition, les hiérarchies sont représentées sous la forme d'arbre. Or les données réelles sont souvent décrites à l'aide de hiérarchies multiples modélisées à l'aide de graphes orientés sans cycle (DAG). L'utilisation de DAGs dans notre proposition nécessitent quelques ajustements afin de garantir la non-redondance de la génération de candidats. Notons que nous nous appuyons sur une stratégie de parcours en profondeur d'abord, nous pouvons donc mettre en œuvre un mécanisme de filtre afin de ne pas parcourir plus d'une fois un élément. Toutefois, nous restons persuader que considérer une hiérarchie par rapport à une autre dans le DAG peut nous permettre d'élaguer plus rapidement l'espace de recherche. Il est nécessaire dans ce cas là de prendre en compte la sémantique de chaque hiérarchie.

La prise en compte des hiérarchies apporte une réelle plus-value dans la qualité des connaissances extraites. C'est pourquoi il est nécessaire de les considérer dans l'extraction de règles séquentielles multidimensionnelles inattendues. Ainsi, au lieu d'instancier les valeurs jokers par les niveaux de granularités les plus fins, l'utilisation de niveaux intermédiaires grâce à la prise en compte des hiérarchies peut permettre de découvrir des règles inattendues plus fines. La qualité des règles inattendues peut également être grandement améliorée par la découverte des règles déclenchantes. C'est à dire retourner à chaque fois les règles inattendues qui ont été découvertes en instanciant le moins possible les valeurs jokers. Ceci nous permettrait de découvrir les facteurs déclenchants des règles inattendues.

Extraction de motifs sous contraintes : un excellent compromis

L'extraction de motifs sous contraintes peut permettre de réunir les deux précédents points. Nous considérons qu'ils sont un excellent compromis entre l'efficacité de l'extraction et la qualité des connaissances extraites. En effet, l'extraction de top k permet de élaguer plus rapidement l'espace de recherche en se basant sur des calculs de bornes supérieures lors du parcours de l'espace de recherche afin de voir si une branche mérite d'être considérée ou non. Les top k permettent aussi de représenter les k « meilleures » connaissances par rapport à un ou plusieurs critères. Ceci n'est pas négligeable lorsque le nombre de motifs extraits est très important puisque le décideur n'est confronté qu'à k motifs.

Toutes les propositions formulées dans le cadre de cette thèse peuvent être adaptées dans un contexte d'extraction de top k. On peut ainsi imaginer extraire les top k motifs séquentiels multidimensionnels, les top k outliers et les top k connaissances inattendues.

Vers d'autres contextes

Les travaux développés dans ce manuscrit mérite d'être utilisés ou adaptés dans d'autres contextes.

L'utilisation d'une autre relation d'ordre que le temps peut être judicieuse. Par exemple, nous pouvons imaginer extraire des motifs séquentiels multidimensionnels avec une relation spatio-temporelle. L'utilisation des motifs séquentiels multidimensionnels en fouille de texte pourrait permettre de prendre en compte de nombreux attributs comme le mot, sa catégorie grammaticale, ses concepts associés, etc.

Nous aimerions également utiliser le temps et la multidimensionnalité pour la prédiction d'événements. En effet, les motifs séquentiels multidimensionnels pourraient être développés afin de d'anticiper des événements et ainsi être utiles dans de nombreuses applications qui nécessitent une très grande réactivité. Par exemple, on peut imaginer détecter une attaque afin que celle-ci n'est aboutie ou anticiper la blessure ou les contre performances d'un athlète. Des approches commencent à prendre en compte le temps pour la prédiction, aucune ne combinent temps et multidimensionnalité.

Ces dernières années, de nombreuses applications se sont développées et nécessitent la fouille de *flots de données*. Les flots de données sont des flux d'informations potentiellement infinis, ce qui rend caduque l'application des approches classiques. En effet, il est impossible de parcourir le flot dans son intégralité au

contraire d'une base de données « statique ». De nouvelles approches se sont développées et proposent une approximation des résultats. Toutefois, les flots de données réels sont multidimensionnels et aucune approche d'extraction de motifs ne permet de gérer cette multidimensionnalité. Par exemple, les paquets TCP/IP circulant sur un réseaux informatiques contiennent de nombreux attributs. Il est primordial de prendre en compte cette multidimensionnalité, c'est pourquoi les motifs séquentiels multidimensionnels méritent d'être adaptés dans ce contexte. Ils permettraient la découverte de tendance de haut niveau en détectant des spécialisation ou des généralisation de comportements généraux au cours du flots.

Vers d'autres types de connaissances

La multidimensionnalité et ses spécificités inhérentes peuvent nous amener à définir de nouveaux types de connaissances. Par exemple, nous avons défini dans le chapitre 2 de la partie II les séquences convergentes et divergentes. De telles séquences peuvent nous conduire à rechercher de nouveaux types de connaissances comme la détection de pandémie, de mode, la découverte d'articles fondateurs, etc.

Pour les règles multidimensionnelles séquentielles inattendues, il serait intéressant de les étendre aux exceptions comme l'approche de [SS96] dans le contexte des règles d'association. Pour cela, il est nécessaire de définir une union efficace entre deux séquences. La duplication des dimensions temporelles dans les dimensions temporelles peut permettre d'effectuer une union efficace entre deux séquences multidimensionnelles.

Enfin, nous pouvons aussi prendre le problème dans l'autre sens : ne plus extraire les séquences fréquentes mais extraire les n-uplets définis sur les dimensions de référence qui maximisent le support de ces séquences.

Bibliographie

[AFGY02] Jay AYRES, Jason FLANNICK, Johannes GEHRKE et Tomi YIU : Sequential pattern mining using a bitmap representation. *In KDD*, pages 429–435. 2002.

[AGS97] Rakesh AGRAWAL, Ashish GUPTA et Sunita SARAWAGI : Modeling multidimensional databases. *In* W. A. GRAY et Per-Åke LARSON, éditeurs : *ICDE*, pages 232–243. IEEE Computer Society, 1997.

[AIS93] Rakesh AGRAWAL, Tomasz IMIELINSKI et Arun N. SWAMI : Mining association rules between sets of items in large databases. *In* Peter BUNEMAN et Sushil JAJODIA, éditeurs : *SIGMOD Conference*, pages 207–216. ACM Press, 1993.

[AS94] R. AGRAWAL et R. SRIKANT : Fast algorithms for mining association rules. *In Proc. of Int. Conf. of Very Large Data Bases (VLDB'94)*, 1994.

[AS95] R. AGRAWAL et R. SRIKANT : Mining sequential patterns. *In* Philip S. YU et Arbee L. P. CHEN, éditeurs : *Proceedings of the Eleventh International Conference on Data Engineering (ICDE'95), March 6-10, 1995, Taipei, Taiwan*, pages 3–14. IEEE Computer Society, 1995.

[AY01] Charu C. AGGARWAL et Philip S. YU : Outlier detection for high dimensional data. *In SIGMOD Conference*, pages 37–46, 2001.

[BBR03] Jean-François BOULICAUT, Artur BYKOWSKI et Christophe RIGOTTI : Free-sets : A condensed representation of boolean data for the approximation of frequency queries. *Data Min. Knowl. Discov.*, 7(1):5–22, 2003.

[BCM04] Fernando BERZAL, Juan-Carlos CUBERO et Nicolás MARÍN : Anomalous association rules. *In IEEE ICDM Workshop Alternative Techniques for Data Mining and Knowledge Discovery.*, 2004.

[BKNS00] Markus M. BREUNIG, Hans-Peter KRIEGEL, Raymond T. NG et Jörg SANDER : Lof : Identifying density-based local outliers. *In* CHEN *et al.* [CNB00], pages 93–104.

[BL94] V. BARNETT et T. LEWIS : *Outliers in Statistical Data*. John Wiley & Sons, 1994.

[BPT97] Elena BARALIS, Stefano PARABOSCHI et Ernest TENIENTE : Materialized views selection in a multidimensional database. *In* Matthias JARKE, Michael J. CAREY, Klaus R. DITTRICH, Frederick H. LOCHOVSKY, Pericles LOUCOPOULOS et Manfred A. JEUSFELD, éditeurs : *VLDB*, pages 156–165. Morgan Kaufmann, 1997.

[BR99] K. Beyer et R. Ramakrishnan : Bottom-up computation of sparse and iceberg cube. *In Proceedings of the 1999 ACM SIGMOD international conference on Management of data*, pages 359–370, 1999.

[CA97] Keith C. C. Chan et Wai-Ho Au : Mining fuzzy association rules. *In CIKM '97 : Proceedings of the sixth international conference on Information and knowledge management*, pages 209–215, New York, NY, USA, 1997. ACM.

[CCL03] A. Casali, R. Cicchetti et L. Lakhal : Cube lattices : A framework for multidimensional data mining. *In Proceedings of the Third SIAM International Conference on Data Mining*, 2003.

[CDF+01] Edith Cohen, Mayur Datar, Shinji Fujiwara, Aristides Gionis, Piotr Indyk, Rajeev Motwani, Jeffrey D. Ullman et Cheng Yang : Finding interesting associations without support pruning. *IEEE Trans. Knowl. Data Eng.*, 13(1):64–78, 2001.

[CG02] Toon Calders et Bart Goethals : Mining all non-derivable frequent itemsets. *In* Tapio Elomaa, Heikki Mannila et Hannu Toivonen, éditeurs : *PKDD*, volume 2431 de *Lecture Notes in Computer Science*, pages 74–85. Springer, 2002.

[CH03] Ruey-Shun Chen et Yi-Chung Hu : A novel method for discovering fuzzy sequential patterns using the simple fuzzy partition method. *J. Am. Soc. Inf. Sci. Technol.*, 54(7):660–670, 2003.

[CH06] Yen-Liang Chen et Tony Cheng Kui Huang : A new approach for discovering fuzzy quantitative sequential patterns in sequence databases. *Fuzzy Sets and Systems*, 157(12): 1641–1661, 2006.

[CHY96] Ming-Syan Chen, Jiawei Han et Philip S. Yu : Data mining : An overview from a database perspective. *IEEE Trans. Knowl. Data Eng.*, 8(6):866–883, 1996.

[CNB00] Weidong Chen, Jeffrey F. Naughton et Philip A. Bernstein, éditeurs. *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, May 16-18, 2000, Dallas, Texas, USA*. ACM, 2000.

[CT98] Luca Cabibbo et Riccardo Torlone : A logical approach to multidimensional databases. *In* Schek *et al.* [SSRA98], pages 183–197.

[CTCH01] R.-S. Chen, G.-H. Tzeng, C.-C. Chen et Y.-C. Hu : Discovery of Fuzzy Sequential Patterns for Fuzzy Partitions in Quantitative Attributes. *In ACS/IEEE International Conference on Computer Systems and Applications (AICCSA)*, pages 144–150, 2001.

[CWC04] Ding-Ying Chiu, Yi-Hung Wu et Arbee L. P. Chen : An efficient algorithm for mining frequent sequences by a new strategy without support counting. *In ICDE*, pages 375–386. IEEE Computer Society, 2004.

[dAFGL04] S. de Amo, D. A. Furtado, A. Giacometti et D. Laurent : An apriori-based approach for first-order temporal pattern mining. *In XIX Simpósio Brasileiro de Bancos de Dados,*

18-20 de Outubro, 2004,Brasília, Distrito Federal, Brasil, Anais/Proceedings, pages 48–62, 2004.

[DHL+01] Guozhu DONG, Jiawei HAN, Joyce M. W. LAM, Jian PEI et Ke WANG : Mining multi-dimensional constrained gradients in data cubes. *In* Peter M. G. APERS, Paolo ATZENI, Stefano CERI, Stefano PARABOSCHI, Kotagiri RAMAMOHANARAO et Richard T. SNODGRASS, éditeurs : *VLDB*, pages 321–330. Morgan Kaufmann, 2001.

[DHP03] D. DUBOIS, E. HÜLLERMEIER et H. PRADE : A note on quality measures for fuzzy association rules. *In Proc. of Int. Fuzzy Systems Association World Congress on Fuzzy Sets and Systems*, LNAI 2715, pages 346–353, 2003.

[DHP06] D. DUBOIS, E. HÜLLERMEIER et H. PRADE : A systematic approach to the assessment of fuzzy association rules. *Data Mining and Knowledge Discovery*, 13:167–192, 2006.

[DS05] Christopher DARTNELL et Jean SALLANTIN : Assisting scientific discovery with an adaptive problem solver. *In Discovery Science*, pages 99–112. 2005.

[EHZ05] Mohammad EL-HAJJ et Osmar R. ZAÏANE : Finding all frequent patterns starting from the closure. *In* Xue LI, Shuliang WANG et Zhao Yang DONG, éditeurs : *ADMA*, volume 3584 de *Lecture Notes in Computer Science*, pages 67–74. Springer, 2005.

[FLT07] Céline FIOT, Anne LAURENT et Maguelonne TEISSEIRE : From crispness to fuzzyness : Three algorithms for soft sequential pattern mining. *IEEE Transactions on Fuzzy Systems*, 15(6):1263–1277, 2007.

[FZFW06] Hongqin FAN, Osmar R. ZAÏANE, Andrew FOSS et Junfeng WU : A nonparametric outlier detection for effectively discovering top-n outliers from engineering data. *In* Wee Keong NG, Masaru KITSUREGAWA, Jianzhong LI et Kuiyu CHANG, éditeurs : *PAKDD*, volume 3918 de *Lecture Notes in Computer Science*, pages 557–566. Springer, 2006.

[Gar59] M GARDNER : *Mathematical games*. Scientific American, 1959.

[GCB+97] Jim GRAY, Surajit CHAUDHURI, Adam BOSWORTH, Andrew LAYMAN, Don REICHART, Murali VENKATRAO, Frank PELLOW et Hamid PIRAHESH : Data cube : A relational aggregation operator generalizing group-by, cross-tab, and sub totals. *Data Min. Knowl. Discov.*, 1(1):29–53, 1997.

[GH06] Liqiang GENG et Howard J. HAMILTON : Interestingness measures for data mining : A survey. *ACM Comput. Surv.*, 38(3), 2006.

[GLSS06] Joydeep GHOSH, Diane LAMBERT, David B. SKILLICORN et Jaideep SRIVASTAVA, éditeurs. *Proceedings of the Sixth SIAM International Conference on Data Mining, April 20-22, 2006, Bethesda, MD, USA*. SIAM, 2006.

[Goe03] B. GOETHALS : Survey on frequent pattern mining, 2003.

[GRS99] Minos N. GAROFALAKIS, Rajeev RASTOGI et Kyuseok SHIM : Spirit : Sequential pattern mining with regular expression constraints. *In* Malcolm P. ATKINSON, Maria E. OR-

LOWSKA, Patrick VALDURIEZ, Stanley B. ZDONIK et Michael L. BRODIE, éditeurs : *VLDB*, pages 223–234. Morgan Kaufmann, 1999.

[Han97] Jiawei HAN : Olap mining : Integration of olap with data mining. *In* Stefano SPACCAPIETRA et Fred J. MARYANSKI, éditeurs : *Data Mining and Reverse Engineering : Searching for Semantics, IFIP TC2/WG2.6 Seventh Conference on Database Semantics (DS-7), October 7-10, 1997, Leysin, Switzerland*, volume 124 de *IFIP Conference Proceedings*, pages 3–20. Chapman & Hall, 1997.

[Haw80] D. HAWKINS : *Identification of Outliers*. Chapman and Hall, London, 1980.

[HF95] Jiawei HAN et Yongjian FU : Discovery of multiple-level association rules from large databases. *In* Umeshwar DAYAL, Peter M. D. GRAY et Shojiro NISHIO, éditeurs : *VLDB*, pages 420–431. Morgan Kaufmann, 1995.

[HF99] Jiawei HAN et Yongjian FU : Mining multiple-level association rules in large databases. *IEEE Trans. Knowl. Data Eng.*, 11(5):798–804, 1999.

[HK00] Jiawei HAN et Micheline KAMBER : *Data Mining : Concepts and Techniques*. Morgan Kaufmann, 2000.

[HK01] Frank HÖPPNER et Frank KLAWONN : A new approach to fuzzy partitioning. *In IFSA World Congress and 20th NAFIPS International Conference, 2001.* IEEE, 2001.

[HLC01] Jia-Lien HSU, Chih-Chin LIU et Arbee L. P. CHEN : Discovering nontrivial repeating patterns in music data. *IEEE Transactions on Multimedia*, 3(3):311–325, 2001.

[HLSL00] Farhad HUSSAIN, Huan LIU, Einoshin SUZUKI et Hongjun LU : Exception rule mining with a relative interestingness measure. *In* Takao TERANO, Huan LIU et Arbee L. P. CHEN, éditeurs : *PAKDD*, volume 1805 de *Lecture Notes in Computer Science*, pages 86–97. Springer, 2000.

[HLW01] T.P. HONG, K.Y. LIN et S.L. WANG : Mining Fuzzy Sequential Patterns from Multiple-Items Transactions. *In Proceedings of the Joint 9th IFSA World Congress and 20th NAFIPS International Conference*, pages 1317–1321, 2001.

[HPY00] Jiawei HAN, Jian PEI et Yiwen YIN : Mining frequent patterns without candidate generation. *In* CHEN *et al.* [CNB00], pages 1–12.

[HTC04] Y.-C. HU, G.-H. TZENG et C.-M. CHEN : Deriving Two-Stage Learning Sequences from Knowledge in Fuzzy Sequential Pattern Mining. *Information Sciences*, 159:69–86, 2004.

[IKA02] Tomasz IMIELINSKI, Leonid KHACHIYAN et Amin ABDULGHANI : Cubegrades : Generalizing association rules. *Data Min. Knowl. Discov.*, 6(3):219–257, 2002.

[JHC+07] Changhai JZHANG, Kongfa HU, Zhuxi CHEN, Ling CHEN et Yisheng DONG : Approxmgmsp : A scalable method of mining approximate multidimensional sequential patterns on distributed system. *In Fourth International Conference on Fuzzy Systems and Knowledge Discovery, 2007. FSKD 2007*, pages 730–734, 2007.

[JLT06] Simon JAILLET, Anne LAURENT et Maguelonne TEISSEIRE : Sequential patterns for text categorization. *Intell. Data Anal.*, 10(3):199–214, 2006.

[KHC97] Micheline KAMBER, Jiawei HAN et Jenny CHIANG : Metarule-guided mining of multi-dimensional association rules using data cubes. *In KDD*, pages 207–210, 1997.

[KLNS04] C. KIM, J.-H. LIM, R. NG et K. SHIM : SQUIRE : Sequential pattern mining with quantities. *In 20th International Conference on Data Engineering (ICDE'04)*, page 827, 2004.

[KN97] Edwin M. KNORR et Raymond T. NG : A unified notion of outliers : Properties and computation. *In KDD*, pages 219–222, 1997.

[KN98] Edwin M. KNORR et Raymond T. NG : Algorithms for mining distance-based outliers in large datasets. *In* Ashish GUPTA, Oded SHMUELI et Jennifer WIDOM, éditeurs : *VLDB*, pages 392–403. Morgan Kaufmann, 1998.

[Kod01] Yves KODRATOFF : Comparing machine learning and knowledge discovery in DataBases : An application to knowledge discovery in texts. *Lecture Notes in Computer Science*, 2049:1–??, 2001.

[Lau03] Anne LAURENT : A new approach for the generation of fuzzy summaries based on fuzzy multidimensional databases. *Intell. Data Anal.*, 7(2):155–177, 2003.

[LB03] Song LIN et Donald E. BROWN : Criminal incident data association using the olap technology. *In* Hsinchun CHEN, Richard MIRANDA, Daniel Dajun ZENG, Chris C. DEMCHAK, Jennifer SCHROEDER et Therani MADHUSUDAN, éditeurs : *ISI*, volume 2665 de *Lecture Notes in Computer Science*, pages 13–26. Springer, 2003.

[LC05] Congnan LUO et Soon Myoung CHUNG : Efficient mining of maximal sequential patterns using multiple samples. *In SDM*, 2005.

[Lev66] V. I. LEVENSHTEIN : Appeared in english as binary codes capable of correcting deletions, insertions, and reversals. *In Soviet Physics Doklady 10 (1966)*, 1966.

[LH07] Xiaolei LI et Jiawei HAN : Mining approximate top-k subspace anomalies in multi-dimensional time-series data. *In* Christoph KOCH, Johannes GEHRKE, Minos N. GAROFALAKIS, Divesh SRIVASTAVA, Karl ABERER, Anand DESHPANDE, Daniela FLORESCU, Chee Yong CHAN, Venkatesh GANTI, Carl-Christian KANNE, Wolfgang KLAS et Erich J. NEUHOLD, éditeurs : *VLDB*, pages 447–458. ACM, 2007.

[LHCM00] Bing LIU, Wynne HSU, Shu CHEN et Yiming MA : Analyzing the subjective interestingness of association rules. *IEEE Intelligent Systems*, 15(5):47–55, 2000.

[LHXS06] Hongyan LIU, Jiawei HAN, Dong XIN et Zheng SHAO : Mining interesting patterns from very high dimensional data : A top-down row enumeration approach. *In* GHOSH *et al.* [GLSS06].

[LL02] Ming-Yen LIN et Suh-Yin LEE : Fast discovery of sequential patterns by memory indexing. *In DaWaK 2002 : Proceedings of the 4th International Conference on Data Warehousing and Knowledge Discovery*, pages 150–160, London, UK, 2002. Springer-Verlag.

[MCP98] Florent MASSEGLIA, Fabienne CATHALA et Pascal PONCELET : The psp approach for mining sequential patterns. *In* Jan M. ZYTKOW et Mohamed QUAFAFOU, éditeurs : *Principles of Data Mining and Knowledge Discovery, Second European Symposium, PKDD '98, Nantes, France, September 23-26, 1998, Proceedings*, volume 1510 de *Lecture Notes in Computer Science*, pages 176–184. Springer, 1998.

[MFT06] H. D. K. MOONESINGHE, Samah FODEH et Pang-Ning TAN : Frequent closed itemset mining using prefix graphs with an efficient flow-based pruning strategy. *In ICDM '06 : Proceedings of the Sixth International Conference on Data Mining*, pages 426–435, Washington, DC, USA, 2006. IEEE Computer Society.

[MH01] Sheng MA et Joseph L. HELLERSTEIN : Mining mutually dependent patterns. *In ICDM '01 : Proceedings of the 2001 IEEE International Conference on Data Mining*, pages 409–416, Washington, DC, USA, 2001. IEEE Computer Society.

[MRBM06] Riadh Ben MESSAOUD, Sabine Loudcher RABASÉDA, Omar BOUSSAID et Rokia MISSAOUI : Enhanced mining of association rules from data cubes. *In* Il-Yeol SONG et Panos VASSILIADIS, éditeurs : *DOLAP*, pages 11–18. ACM, 2006.

[MTV97] H. MANNILA, H. TOIVONEN et A. Inkeri VERKAMO : Discovery of frequent episodes in event sequences. *Data Min. Knowl. Discov.*, 1(3):259–289, 1997.

[NJ03] Svetlozar NESTOROV et Nenad JUKIC : Ad-hoc association-rule mining within the data warehouse. *In HICSS*, page 232, 2003.

[PBTL99] Nicolas PASQUIER, Yves BASTIDE, Rafik TAOUIL et Lotfi LAKHAL : Discovering frequent closed itemsets for association rules. *In* Catriel BEERI et Peter BUNEMAN, éditeurs : *ICDT*, volume 1540 de *Lecture Notes in Computer Science*, pages 398–416. Springer, 1999.

[PCT+03] Feng PAN, Gao CONG, Anthony K. H. TUNG, Jiong YANG et Mohammed J. ZAKI : Carpenter : finding closed patterns in long biological datasets. *In KDD '03 : Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 637–642, New York, NY, USA, 2003. ACM.

[Pei02] Jian PEI : *Pattern-growth Methods for Frequent Pattern Mining*. Thèse de doctorat, School of Computing Science, Simon Fraser University, Canada, 2002.

[PHM00] Jian PEI, Jiawei HAN et Runying MAO : Closet : An efficient algorithm for mining frequent closed itemsets. *In ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, pages 21–30, 2000.

[PHMA+01] Jian PEI, Jiawei HAN, Behzad MORTAZAVI-ASL, Helen PINTO, Qiming CHEN, Umeshwar DAYAL et Meichun HSU : Prefixspan : Mining sequential patterns by prefix-projected growth. *In ICDE*, pages 215–224. IEEE Computer Society, 2001.

[PHMA+04] J. Pei, J. Han, B. Mortazavi-Asl, J. Wang, H. Pinto, Q. Chen, U. Dayal et M.-C. Hsu : Mining sequential patterns by pattern-growth : The prefixspan approach. *IEEE Transactions on Knowledge and Data Engineering*, 16(10), 2004.

[PHP+01] H. Pinto, J. Han, J. Pei, K. Wang, Q. Chen et U. Dayal : Multi-dimensional sequential pattern mining. *In CIKM*, pages 81–88. ACM, 2001.

[PTCX04] Feng Pan, Anthony K. H. Tung, Gao Cong et Xin Xu : Cobbler : Combining column and row enumeration for closed pattern discovery. *In SSDBM*, pages 21–30. IEEE Computer Society, 2004.

[RKK07] Sherif Rashad, Mehmed M. Kantardzic et Anup Kumar : Msp-cacrr : Multidimensional sequential patterns based call admission control and resource reservation for next-generation wireless cellular networks. *In CIDM*, pages 552–559. IEEE, 2007.

[RRS00] Sridhar Ramaswamy, Rajeev Rastogi et Kyuseok Shim : Efficient algorithms for mining outliers from large data sets. *In* Chen *et al.* [CNB00], pages 427–438.

[SA96a] R. Srikant et R. Agrawal : Mining sequential patterns : Generalizations and performance improvements. *In* Peter Apers, Mokrane Bouzeghoub et Georges Gardarin, éditeurs : *Advances in Database Technology - EDBT'96, 5th International Conference on Extending Database Technology, Avignon, France, March 25-29, 1996, Proceedings*, volume 1057 de *Lecture Notes in Computer Science*, pages 3–17. Springer, 1996.

[SA96b] Ramakrishnan Srikant et Rakesh Agrawal : Mining quantitative association rules in large relational tables. *SIGMOD Rec.*, 25(2):1–12, 1996.

[Sah99] Sigal Sahar : Interestingness via what is not interesting. *In Knowledge Discovery and Data Mining*, pages 332–336, 1999.

[SAM98] Sunita Sarawagi, Rakesh Agrawal et Nimrod Megiddo : Discovery-driven exploration of olap data cubes. *In* Schek *et al.* [SSRA98], pages 168–182.

[SBI06] Panida Songram, Veera Boonjing et Sarun Intakosum : Closed multidimensional sequential pattern mining. *In ITNG*, pages 512–517. IEEE Computer Society, 2006.

[SCA06] Pei Sun, Sanjay Chawla et Bavani Arunasalam : Mining for outliers in sequential databases. *In* Ghosh *et al.* [GLSS06].

[SG92] Padhraic Smyth et Rodney M. Goodman : An information theoretic approach to rule induction from databases. *IEEE Trans. Knowl. Data Eng.*, 4(4):301–316, 1992.

[SG05] R. B. V. Subramanyam et A. Goswami : A fuzzy data mining algorithm for incremental mining of quantitative sequential patterns. *International Journal of Uncertainty Fuzziness Knowledge-Based Systems*, 13(6):633–652, 2005.

[SS96] Einoshin Suzuki et Masamichi Shimura : Exceptional knowledge discovery in databases based on information theory. *In KDD*, pages 275–278, 1996.

[SSRA98] Hans-Jörg SCHEK, Fèlix SALTOR, Isidro RAMOS et Gustavo ALONSO, éditeurs. *Advances in Database Technology - EDBT'98, 6th International Conference on Extending Database Technology, Valencia, Spain, March 23-27, 1998, Proceedings*, volume 1377 de *Lecture Notes in Computer Science*. Springer, 1998.

[ST96] Abraham SILBERSCHATZ et Alexander TUZHILIN : What makes patterns interesting in knowledge discovery systems. *IEEE Trans. Knowl. Data Eng.*, 8(6):1996, 1996.

[Suz99] Einoshin SUZUKI : Scheduled discovery of exception rules. *In DS '99 : Proceedings of the Second International Conference on Discovery Science*, pages 184–195, London, UK, 1999. Springer-Verlag.

[Suz02a] Einoshin SUZUKI : In pursuit of interesting patterns with undirected discovery of exception rules. *In* Setsuo ARIKAWA et Ayumi SHINOHARA, éditeurs : *Progress in Discovery Science*, volume 2281 de *Lecture Notes in Computer Science*, pages 504–517. Springer, 2002.

[Suz02b] Einoshin SUZUKI : Undirected discovery of interesting exception rules. *IJPRAI*, 16(8):1065–1086, 2002.

[SVA97] Ramakrishnan SRIKANT, Quoc VU et Rakesh AGRAWAL : Mining association rules with item constraints. *In KDD*, pages 67–73, 1997.

[SZ05a] Jerzy STEFANOWSKI et Radoslaw ZIEMBINSKI : Mining context based sequential patterns. *In* Piotr S. SZCZEPANIAK, Janusz KACPRZYK et Adam NIEWIADOMSKI, éditeurs : *AWIC*, volume 3528 de *Lecture Notes in Computer Science*, pages 401–407. Springer, 2005.

[SZ05b] Einoshin SUZUKI et Jan M. ZYTKOW : Unified algorithm for undirected discovery of exception rules. *International Journal of Intelligent Systems*, 20(7):673–691, 2005.

[TC90] Henry S. TENG et Kaihu CHEN : Adaptive real-time anomaly detection using inductively generated sequential patterns. *sp*, 00:278, 1990.

[TKS04] Pang-Ning TAN, Vipin KUMAR et Jaideep SRIVASTAVA : Selecting the right objective measure for association analysis. *Inf. Syst.*, 29(4):293–313, 2004.

[TT04] Haorianto Cokrowijoyo TJIOE et David TANIAR : Mining hybrid association rules in data warehouses. *In* Stéphane BRESSAN, David TANIAR, Gabriele KOTSIS et Ismail Khalil IBRAHIM, éditeurs : *iiWAS*, volume 183 de *books@ocg.at*. Austrian Computer Society, 2004.

[TTM04] D. TANASA, B. TROUSSE et Florent MASSEGLIA : *Mesures de l'internet*, chapitre Fouille de données appliquées au logs web : état de l'art sur le Web Usage Mining, pages 126–143. édition Les Canadiens en Europe, 2004.

[UAUA04] Takeaki UNO, Tatsuya ASAI, Yuzo UCHIDA et Hiroki ARIMURA : An efficient algorithm for enumerating closed patterns in transaction databases. *In* Einoshin SUZUKI et Setsuo ARIKAWA, éditeurs : *Discovery Science*, volume 3245 de *Lecture Notes in Computer Science*, pages 16–31. Springer, 2004.

[Vas98] Panos VASSILIADIS : Modeling multidimensional databases, cubes and cube operations. *In* Maurizio RAFANELLI et Matthias JARKE, éditeurs : *SSDBM*, pages 53–62. IEEE Computer Society, 1998.

[WHL07] Jianyong WANG, Jiawei HAN et Chun LI : Frequent closed sequence mining without candidate maintenance. *IEEE Trans. Knowl. Data Eng.*, 19(8):1042–1056, 2007.

[WK04] Jianyong WANG et George KARYPIS : Bamboo : Accelerating closed itemset mining by deeply pushing the length-decreasing support constraint. *In* Michael W. BERRY, Umeshwar DAYAL, Chandrika KAMATH et David B. SKILLICORN, éditeurs : *SDM*. SIAM, 2004.

[YC05] C.-C. YU et Y.-L. CHEN : Mining sequential patterns from multidimensional sequence data. *IEEE Transactions on Knowledge and Data Engineering*, 17(1):136–140, 2005.

[YHA03] Xifeng YAN, Jiawei HAN et Ramin AFSHAR : Clospan : Mining closed sequential patterns in large databases. *In* Daniel BARBARÁ et Chandrika KAMATH, éditeurs : *SDM*. SIAM, 2003.

[YHN06] S. Ben YAHIA, T. HAMROUNI et E. Mephu NGUIFO : Frequent closed itemset based algorithms : a thorough structural and analytical survey. *SIGKDD Explor. Newsl.*, 8(1):93–104, 2006.

[YKW06] Zhenglu YANG, Masaru KITSUREGAWA et Yitong WANG : Paid : Mining sequential patterns by passed item deduction in large databases. *In IDEAS*, pages 113–120. IEEE Computer Society, 2006.

[YWK07] Zhenglu YANG, Yitong WANG et Masaru KITSUREGAWA : Lapin : Effective sequential pattern mining algorithms by last position induction for dense databases. *In* Kotagiri RAMAMOHANARAO, P. Radha KRISHNA, Mukesh K. MOHANIA et Ekawit NANTAJEEWARAWAT, éditeurs : *DASFAA*, volume 4443 de *Lecture Notes in Computer Science*, pages 1020–1023. Springer, 2007.

[YWY04] Jiong YANG, Wei WANG et Philip S. YU : Mining surprising periodic patterns. *Data Min. Knowl. Discov.*, 9(2):189–216, 2004.

[YWYH02] Jiong YANG, Wei WANG, Philip S. YU et Jiawei HAN : Mining long sequential patterns in a noisy environment. *In SIGMOD '02 : Proceedings of the 2002 ACM SIGMOD international conference on Management of data*, pages 406–417, New York, NY, USA, 2002. ACM Press.

[Zak01] Mohammed Javeed ZAKI : Spade : An efficient algorithm for mining frequent sequences. *Machine Learning*, 42(1/2):31–60, 2001.

[ZH02] Mohammed Javeed ZAKI et Ching-Jiu HSIAO : Charm : An efficient algorithm for closed itemset mining. *In* Robert L. GROSSMAN, Jiawei HAN, Vipin KUMAR, Heikki MANNILA et Rajeev MOTWANI, éditeurs : *SDM*. SIAM, 2002.

[Zhu98] Hua ZHU : On-line analytical mining of association rules. Mémoire de Master, Burnaby, British Columbia V5A 1S6, Canada, 1998.

[ZYH+07] Feida ZHU, Xifeng YAN, Jiawei HAN, Philip S. YU et Hong CHENG : Mining colossal frequent patterns by core pattern fusion. *In ICDE*, pages 706–715. IEEE, 2007.

[ZYO03] Ning ZHONG, Yiyu YAO et Muneaki OHSHIMA : Peculiarity oriented multidatabase mining. *IEEE Trans. Knowl. Data Eng.*, 15(4):952–960, 2003.

Articles publiés dans le cadre de cette thèse

Revues internationales avec comité de lecture

- M. Plantevit, A. Laurent et M. Teisseire. OLAP-Sequential Mining : Summarizing Trends from Historical Multidimensional Data using Closed Multidimensional Sequential Patterns. *Annals of Information Systems, special issue in New Trends in Data Warehousing and Data Analysis*, à paraître (2008).

Revue nationale avec comité de lecture

- M. Plantevit. Motifs Séquentiels Multidimensionnels : Concepts et Techniques. *Revue I3, numéro spécial*, 2007.

Conférences internationales avec comité de lecture

- M. Plantevit, A. Laurent et M. Teisseire. Up and Down : Mining Multidimensional Sequential Patterns Using Hierarchies. Dans les actes de *International Conference on Data Warehousing and Knowledge Discovery (DaWaK'08)* , Septembre 2008.

- C. Raïssi et M. Plantevit. Mining Multidimensional Sequential Patterns over Data Streams. Dans les actes de *International Conference on Data Warehousing and Knowledge Discovery (DaWaK'08)* , Septembre 2008.

- M. Plantevit, S. Goutier, F. Guisnel, A. Laurent et M. Teisseire. Mining Unexpected Multidimensional Rules. Dans les actes de *ACM DOLAP (DOLAP'07)* , Novembre 2007.

- M. Plantevit, A. Laurent et M. Teisseire. HYPE : Mining Hierarchical Sequentiel Patterns. Dans les actes de *ACM DOLAP (DOLAP'06)* , Novembre 2006.

- M. Plantevit, Y.W Choong, A. Laurent, D. Laurent et M. Teisseire. M2SP : Mining Sequential Patterns among Several Dimensions. Dans les actes de *Principles et Practice of Knowledge Discovery in Databases Conference (PKDD'05)*, Octobre 2005.

Conférences nationales avec comité de lecture

- M. Plantevit,, A. Laurent et M. Teisseire. Fouille de Données Multidimensionnelles : Différentes Stratégies pour Prendre en Compte la Mesure.. Dans les actes de *Entrepôts de Données et Analyse en ligne (EDA'08)*, Juin 2008.

- M. Plantevit,, A. Laurent et M. Teisseire. Extraction de Motifs Sequentiels Multidimensionnels Clos Sans Gestion d'Ensemble de Candidats Dans les actes des *journées d'Extraction et Gestion des Connaissances (EGC'08)*, Janvier 2008.

- C. Fiot, M. Plantevit et D. Jouve. Quelle Partition pour les Motifs Séquentiels Multidimensionnels ? Dans les actes des *Rencontres Francophones sur la Logique Floue et ses Applications (LFA'07)*, Novembre 2007.

- M. Plantevit,, A. Laurent et M. Teisseire. Extraction d'outliers dans des cubes de données : une aide à la navigation. Dans les actes de *Entrepôts de Données et Analyse en ligne (EDA'07)*, Juin 2007.

- M. Plantevit, A. Laurent et M. Teisseire. Motifs Séquentiels Multidimensionnels Convergents et Divergents Dans les actes des *journées d'Extraction et Gestion des Connaissances (EGC'07)*, Janvier 2007.

- M. Plantevit,, A. Laurent et M. Teisseire. HYPE : Prise en compte des hiérarchies lors de l'extraction de motifs séquentiels multidimensionnels. Dans les actes de *Entrepôts de Données et Analyse en ligne (EDA'06)*, Juin 2006.

- M. Plantevit, A. Laurent et M. Teisseire. Motifs Séquentiels Multidimensionnels Etoilés. Dans les actes des *Journées des Bases de Données Avancées (BDA'05)*, Novembre 2005.

Annexes

Annexe A

Description des données réelles utilisées

Dans cette annexe, nous décrivons brièvement les données issues des cubes de données d'EDF (Électricité De France). Nous disposons de ces données dans le cadre d'une collaboration scientifique entre EDF Recherche & Développement et l'équipe de fouille de données (Tatoo) du LIRMM qui vise à développer des outils efficaces de détections de comportements temporels atypiques dans des données OLAP. Ces données sont confidentielles, nous ne pouvons donc pas trop les décrire précisément.

Nous considérons la table simplifiées (Tab. A.1), décrivant l'activité marketing de nombreux clients résidentiels d'EDF (environ 30 million de clients résidentiels). Le schéma simplifié de la base de données est MARKETING_OFFER (#OFFER, OFFER, STATE_OF_OFFER, HEATING, GEOGRAPHY, SUPPORT_OF_OFFER, TIME)

Un n-uplet de Tab. A.1 décrit une offre (OFFER à pour différents statuts STATE_OF_OFFER tels que *current* (*FLUX*) ou *closed* (*STOCK*) et différentes caractéristiques des consommateurs :

- le type de chauffage principal (HEATING),
- la localisation géographique des consommateurs par rapport aux centres et sous-centres d'EDF (GEOGRAPHY),

#OFFER	OFFER	STATE_OF_ OFFER	HEATING	GEOGRAPHY	SUPPORT_OF_ OFFER	TIME
1	BIEN	Flux	Electrical	Bordeaux	Phoning	Jan 2003
2	BIEN	Flux	Gas	Montpellier	Mailing	Mar 2005
3	RENO	Stock	Electrical	Lyon	Phoning	Nov 2004
4	DCS	Flux	Gas	Lyon	Phoning	Feb 2003
5	RENO	Stock	Fuel	Paris	Mailing	Jan 2004
…	…	…	…	…	…	…

TAB. A.1 – Exemple de données

- le support de démarchage des consommateurs (SUPPORT_OF_OFFER) tels que *mailing* ou *phoning*,
- la date des opérations (TIME).

Annexe B

Base de données multidimensionnelles

Dans cette annexe, nous présentons un bref aperçu des bases de données multidimensionnelles et les différentes approches de matérialisation d'un entrepôt de données.

Généralités

De manière générale, les données multidimensionnelles sont stockées dans des **cubes** sur lesquels plusieurs opérations de manipulation sont définies. Les cubes sont organisés en plusieurs dimensions qui peuvent être elles-même organisées selon plusieurs niveaux de hiérarchie.

Les cubes sont eux-mêmes composés d'un ensemble de **cellules**.

La **mesure** est la valeur contenue dans une cellule du cube[1], associée aux valeurs prises sur toutes les dimensions composant le cube comme l'illustre la figure B.1. Il peut également exister des hiérarchies sur les dimensions.

La **table de faits** est la matérialisation d'une association entre n entités. Les **tables de faits** représentent des associations dont l'existence d'une occurence dépend de l'existence des occurences correspondantes dans les tables dimensionnelles, c'est à dire la table de faits contient l'ensemble des mesures correspondant aux informations de l'activité à analyser. Mais il est nécessaire de rapeller que certaines tables de faits peuvent contenir aucun attribut et ne représentent que des liaisons entre les tables. Tous les éléments qui pointent sur la table de faits sont liés à une sémantique exprimable par une phrase.

Les opérations sur les cubes favorisent la navigation à l'intérieur du cube et sa visualisation par l'utilisateur. Deux types d'opérations sont définies :

D'une part, des opérations permettent de modifier la **structure** du cube telles que la rotation (*rotate*), l'inversion de deux dimensions (*Switch*), la décomposition (*split*) et la concaténation de cubes (*Push*).

[1] Ainsi dans un cube de données, la valeur nulle est associée à des cellules non présentes dans *la table des faits*.

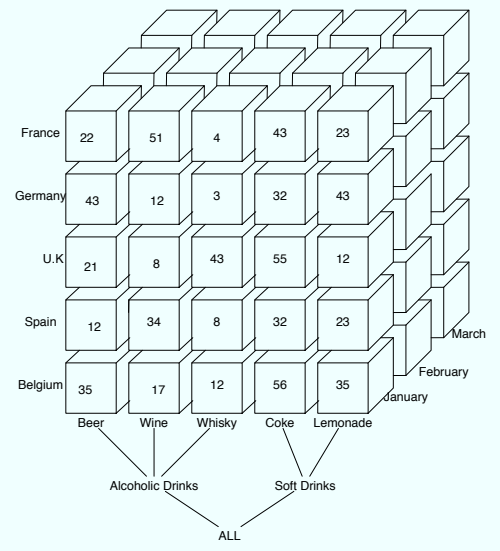


FIG. B.1 – Un cube de données

D'autre part, des opérations permettent la modification du **contenu** des cubes telles que les opérations de généralisation (*resp. spécialisation*) permettant de passer à un niveau de hierarchie supérieur (*resp. inférieur*) (Roll-up (*resp. Roll-down*)), la restriction du cube par rapport à un ensemble de dimensions (*Slice*) et la projection du cube sur un ensemble de dimensions (*Dice*)

A l'heure actuelle deux approches sont proposées pour la création d'un entrepôt de donnéees. Les modèles multidimensionnels sont construits soit sur un modèle relationnel, soit sur un modèle physique réellement multidimensionnel.

- Le modèle ROLAP (Relational OLAP) adopte donc un point de vue relationnel au niveau du stockage des données multidimensionnelles. Une telle approche s'avère très performante et robuste face aux gros volumes de données. Mais le temps de réponse à certaines requêtes peut être un handicap.
- Le modèle MOLAP (Multidimensional OLAP) permet la matérialisation des données sous forme multidimensionnelle, et ainsi optimise le temps de réponse aux requêtes. Cependant, contrairement à l'approche précédente, ces systèmes ne sont pas robuste face à de gros volumes de données, et particulièrement coûteux en terme d'espace de stockage.

Le Modèle ROLAP

Le modèle ROLAP permet d'aborder l'approche OLAP avec un point de vue relationnel en incluant :

- une vision multidimensionnelle des données
- le calcul des données agrégées
- les opérations de navigation des données organisées en différents niveaux de hiérarchie.
- l'extension des requêtes SQL aux besoins d'OLAP.

De tels systèmes sont spécifiquement conçus pour une représentation multidimensionnelle, et sont organisés autour de deux types de données :

1. Les mesures (valeurs numériques) organisées dans *des tables de faits*
2. Les dimensions stockées dans des tables satellites.

Ainsi, les dimensions sont jointes aux tables de faits dans *un schéma en étoile* comme illustré dans la figure B.2. Cependant, le schéma en étoile s'applique sur des données *non normalisées* et provoque ainsi le stockage de redondances.

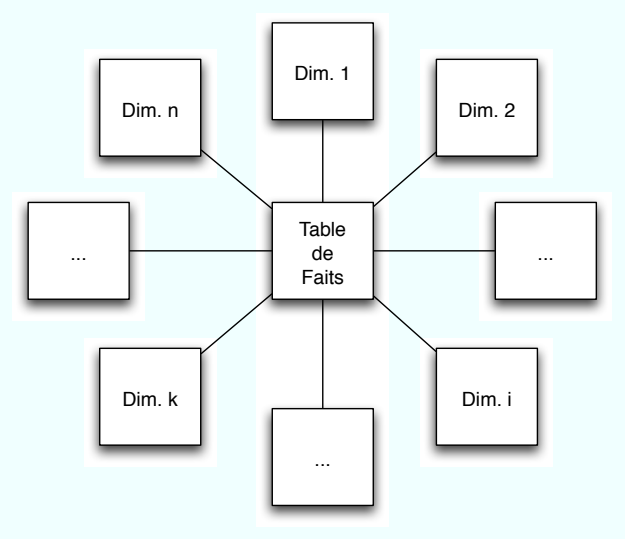


FIG. B.2 – Schéma en étoile

Ainsi, afin d'optimiser l'espace de stockage et éviter les redondances dans le stockage des données, *les schémas en flocon* (figure B.3) basés sur des dépendances fonctionnelles entre les dimensions sont proposés et s'avèrent particulièrement efficaces dans le traitement de données organisées en plusieurs niveaux de hiérarchie. Ils s'appuient sur des données normalisées (*3FN*). Ces schémas facilitent l'interprétation des données, et offrent ainsi une meilleure performance des requêtes.

Plusieurs travaux developpent un point de vue relationnel dans le stockage des données multidimensionnelles [BPT97],[GCB+97].

Le Modèle MOLAP

Les modèles MOLAP [AGS97] [Vas98] [CT98] stockent les données à l'aide de tableaux multidimensionnels. Les données sont donc matérialisées sous forme multidimensionnelle. Ces modèles sont ainsi très performants pour les temps de réponse aux requêtes. De plus, les opérations sur les hiérarchies ne nécessitent aucune opération coûteuse de jointure (contrairement au modèle ROLAP en étoile).

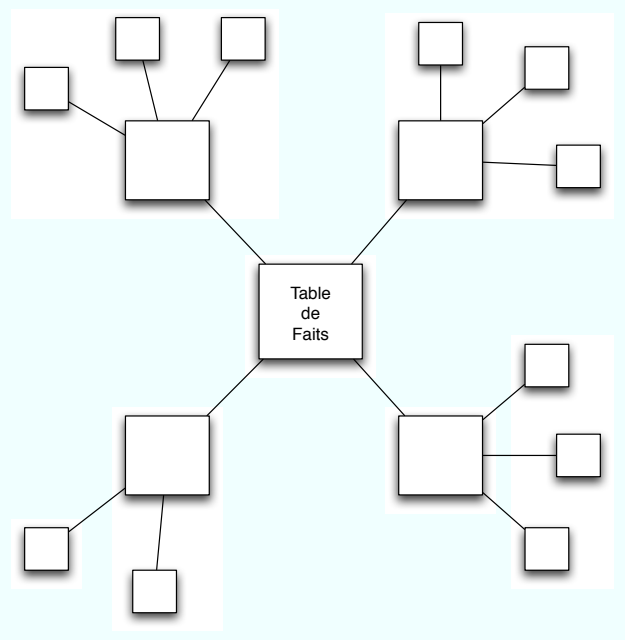


FIG. B.3 – Schéma en flocon

Cependant, ces modèles ne présentent pas que des avantages. Les mises à jour sont très coûteuses. De surcroît, à l'heure actuelle, il n'existe aucun modèle permettant de représenter ou d'interroger les données stockées qui fasse référence.

OLAP & utilisateur

Dans le processus décisionnel OLAP, l'utilisateur a deux possiblités :

1. Soit à l'aide des différentes opérations de manipulation de cubes de données, il va effectuer des requêtes ciblées afin de naviguer à l'intérieur du cube et ainsi, au fil des requêtes, découvrir des informations, et tenter de dégager des tendances.
2. Soit utiliser des outils d'extraction d'information spécifique à OLAP (*OLAP Mining* [Han97]), afin d'obtenir, *de manière automatique*, des corrélations entre des blocs du cube de données et offrir ainsi à l'utilisateur une aide à la décision efficace.

Extraction De Motifs Séquentiels Dans Des Données Multidimensionnelles

L'extraction de motifs séquentiels est devenue, depuis son introduction, une technique majeure du domaine de la fouille de données avec de nombreuses applications (analyse du comportement des consommateurs, bioinformatique, sécurité, musique, etc.). Les motifs séquentiels permettent la découverte de corrélations entre événements en fonction de leurs chronologies d'apparition. Il existe de nombreux algorithmes permettant l'extraction de tels motifs. Toutefois, ces propositions ne prennent en compte qu'une seule dimension d'analyse (e.g le produit dans les applications de type étude des achats des consommateurs) alors que la plupart des données réelles sont multidimensionnelles par nature. Dans ce manuscrit, nous définissons les motifs séquentiels multidimensionnels afin de prendre en compte les spécificités inhérentes aux bases de données multidimensionnelles (plusieurs dimensions, hiérarchies, valeurs agrégées). Nous définissons des algorithmes permettant l'extraction de motifs séquentiels multidimensionnels en tenant compte des ces spécificités. Des expérimentations menées sur des données synthétiques et sur des données réelles sont rapportées et montrent l'intérêt de nos propositions. Nous nous interessons également à l'extraction de comportements temporels atypiques dans des données multidimensionnelles. Nous montrons qu'il peut y avoir plusieurs interprétations d'un comportement atypique (fait ou connaissance). En fonction de chaque interprétation, nous proposons une méthode d'extraction de tels comportements. Ces méthodes sont également validées par des expérimentations sur des données réelles.

Mining Sequential Patterns In Multidimensional Data

Sequential pattern mining is a key technique of data mining with broad applications (user behavior analysis, bioinformatic, security, music, etc.). Sequential pattern mining aims at discovering correlations among events through time. There exist many algorithms to discover such patterns. However, these approaches only take one dimension into account (e.g. product dimension in customer market basket problem analysis) whereas data are multidimensional in nature. In this thesis, we define multidimensional sequential patterns to take the specificity of multidimensional databases (several dimensions, hierarchies, aggregated value). We define algorithms that allow the discovery of such patterns by handling this specificity. Some experiments on both synthetic and real data are reported and show the interest of our proposals. We also focus on the discovery of atypical behavior. We show that there are several interpretations of an atypical behavior (fact or knowledge). According to each interpretation, we propose an approach to discover such behaviors. These approaches are also validated with experiments on real data.

Mots-clés : Motifs séquentiels multidimensionnels, hiérarchies, mesures, motifs clos, connaissances inattendues, outliers.

Keywords : Multidimensional sequential patterns, hierarchies, measure, closed patterns, unexpected knowledge, outlier .

Discipline : Informatique

Laboratoire : Laboratoire d'Informatique de Robotique et de Micro-électronique de Montpellier
Université Montpellier II - CNRS (UMR 5506)
161 rue Ada - 34392 Montpellier cedex 5 - France