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

Chedy Raïssi

le 15 Juillet 2008

Extraction de séquences fréquentes : des bases de données statiques aux flots de données

Jury

Jean-François Boulicaut, Professeur, INSA Lyon, ... Rapporteur
Osmar Zaïane, Professeur, University of Alberta, Canada, ... Rapporteur
Toon Calders, Professeur, Eindhoven Technical University, Pays-Bas, ... Examinateur
Bruno Cremilleux, Professeur, GREYC Caen, ... Examinateur
Anne Laurent , Maître de Conférences, Université Montpellier 2, ... Examinateur
Jian Pei, Professeur, Simon Fraser University, Canada, ... Examinateur
Maguelonne Teisseire , Maître de Conférences, Université Montpellier 2, ... Examinateur
Pascal Poncelet , Professeur, Ecole des Mines d'Alès, ... Directeur de thèse

Le présent manuscrit est le fruit d'une recherche menée le long de plusieurs années, plus précisément de 2005 à 2008, auprès du laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier et de l'Ecole des Mines d'Alès. Une thèse est avant tout une longue suite de collaborations et de découvertes scientifiques. Je tiens donc ici à remercier chaleureusement toutes les personnes avec qui j'ai pu travailler, discuter, rire, m'énerver et encore retravailler au cours de ces quelques années.

Ma dette de reconnaissance va en premier lieu à mon directeur de thèse, Pascal Poncelet, pour l'intérêt qu'il a manifesté à mon égard et ses encouragements. Sans son appui constant, sa disponibilité et sa sagacité bienveillante, je n'aurai jamais réussi à mener à bon terme ce travail. *Merci du fond du coeur Pascal.*

J'adresse aussi tous mes remerciements à Jean-François Boulicaut et Osmar Zaïane pour avoir accepté d'être rapporteurs de cette thèse et pour le temps précieux qu'ils ont consacré à cette tâche. Je remercie également Bruno Crémilleux et Jian Pei pour avoir accepté de faire partie de mon jury lors de la soutenance de ce mémoire.

Je tiens à remercier tout particulièrement Toon Calders pour son intelligence de tous les domaines, son exigence, son humilité et son indépendance d'esprit, pour être quelqu'un capable également d'une grande attention, une personne extraordinaire dont la rencontre et la confiance m'ont très profondément marqué.

Mes remerciements les plus chaleureux vont également à Maguelonne Teisseire et Anne Laurent, responsables de l'équipe TATOO[1], qui m'ont épaulées depuis la maîtrise et qui n'ont jamais manqué d'intérêt pour mes travaux.

Je tiens aussi à remercier tous les anciens thésards qui ont contribué, avec gentillesse et toute leur expertise à l'accomplissement de mes travaux : Lylia Abrouk, Abdelkader Gouaich, Simon Jaillet, Mehdi Yousfi-Monod, Luc Fabresse, Jérôme Chapelle, Christopher Dartnell et tous les autres que j'ai oublié.

Cette thèse n'aurait pas pu s'écrire sans l'appui moral de ma famille et de mes amis. Je les remercie tous et souhaite que la lecture qui s'offre à leur curiosité leur procure la satisfaction qu'ils espéraient.

[1] C'est bel et bien le nom de l'équipe...

Je garde enfin mes remerciements amoureux pour ma compagne Emilie, qui m'a soutenu au quotidien durant ces années de thèse.

A mes très chers parents.

J'espère qu'ils trouveront dans ce travail toute ma reconnaissance et tout mon amour.

La destination du chercheur dépend de la route qu'il suit.

Ibn Al-'Arabi

Sommaire

Table des figures

Liste des Algorithmes

Chapitre 1

Introduction

Lorsque les premiers pionniers de l'informatique se lancèrent dans la construction d'ordinateurs tels que le PDP-1, l'IBM 1401 ou l'Altair 8800, peu d'entre eux, pour ne pas dire aucun, n'imaginaient un jour l'essor fantastique que ce domaine allait prendre. Tout au plus quelques auteurs de science-fiction comme I. Asimov ou F. Herbert, imaginaient dans un futur très lointain des machines de calcul générant de l'information et analysant des connaissances pour le bien-être de l'humanité. Comme souvent au cours de l'Histoire, la réalité a pris le dessus sur l'imaginaire des hommes et nous sommes bel et bien dans une ère dite *numérique*, où les informations sont générées à la volée par des machines, envoyées, capturées, partagées, commentées mais ... difficilement *analysées*.

Il est reconnu aujourd'hui que l'être humain est généralement noyé sous une profusion d'informations et que sa capacité d'analyse n'est plus capable de faire face au volume sans cesse croissant de données. C'est dans ce contexte qu'est né le processus de *l'Extraction de Connaissance dans les bases de Données* (ECD ou KDD[1] en anglais). Ce domaine accentue l'idée que l'on n'a pas besoin de tout savoir. *On a seulement besoin de savoir où trouver ce que l'on cherche quand cela est nécessaire*. De plus, on souhaite pouvoir procéder à travers les différentes étapes de l'ECD (voir Figure 1.1) à une transformation similaire à celle de la matière première vers un produit fini : ainsi, on veut passer d'un grand volume d'informations à un petit ensemble de *connaissances "utilisables"* humainement. Pourtant, le processus d'ECD n'est pas un processus monolithique et univoque au cours duquel il s'agirait d'appliquer un principe général à tous les types de données stockées ou récupérées. Ainsi, une des étapes de ce processus qu'est la fouille de données et les types de *pépites* que l'on veut trouver peuvent se dériver sous plusieurs formes telles que : la *segmentation* [JD88], la *classification* [WK91], *l'extraction d'itemset* et de *règles d'associations* [AIS93, AS94], l'extraction de structures plus complexes telles que les *épisodes, les graphes* [MTV97, WM03] ou comme dans le cadre de cette thèse *l'extraction de motifs séquentiels* [Sri95, SA96].

[1] Knowledge Discovery in Databases.

Le contexte dans lequel les travaux d'extractions ont été définis ces dernières années considérait que les données, sur lesquelles la fouille était réalisée, étaient disponibles dans des bases de données statiques. Aujourd'hui, suite au développement de nouvelles applications, nous devons faire face à de nouveaux modèles dans lesquels les données sont disponibles sous la forme de flots. Une question se pose alors : *quid des approches d'extraction de connaissances traditionnelles ?*

La suite de ce chapitre est organisée de la manière suivante. Dans la section 1 nous présentons brièvement la problématique des motifs séquentiels qui sont au cœur de ce mémoire. L'objectif de la section 2 est de présenter le nouveau contexte dans lequel nous souhaitons extraire de la connaissance : les flots de données. Au cours de la section 3, nous présentons nos motivations ainsi que les principales contributions qui seront détaillées dans ce mémoire. Enfin la section 4 propose le plan du mémoire.

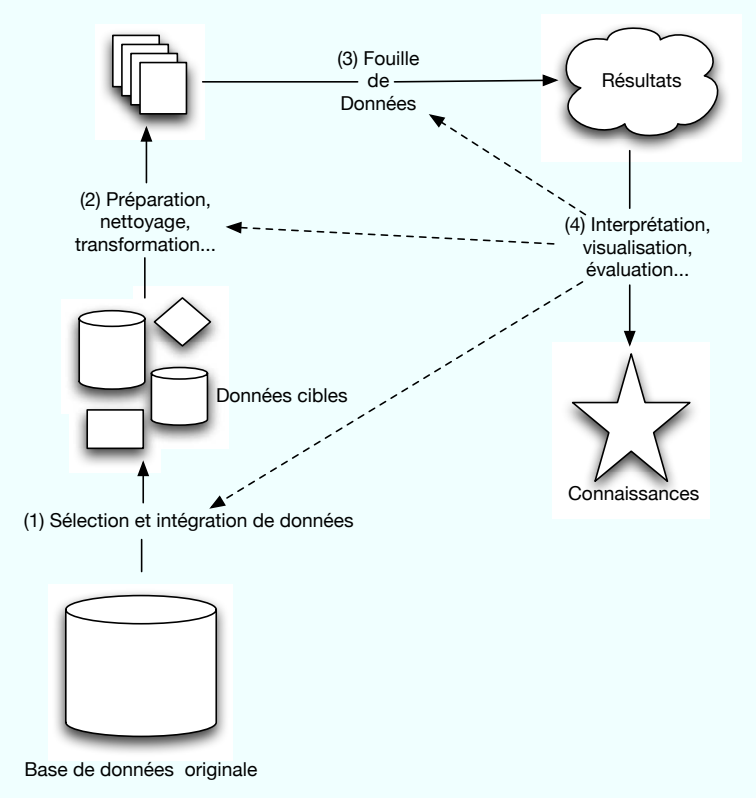


Fig. 1.1: Les différentes étapes de l'"Extraction de Connaissance dans les Bases de Données

1 Motifs séquentiels

La problématique de l'extraction de motifs séquentiels peut être vue comme une extension de la problématique de l'extraction d'itemsets et de règles d'associations. En effet, dans ce cadre, la

dimension temporelle n'est pas prise en compte alors que pour les motifs séquentiels elle occupe une place centrale.

La motivation derrière l'extraction des itemsets et des règles d'associations vient historiquement d'un besoin d'analyse des données issues de transactions de supermarché (appelé traditionnellement le problème du *panier de la ménagère*) afin d'extraire des règles qui permettent de mettre en exergue des corrélations entre les différents achats des clients *dans une même transaction*. L'extraction d'itemsets est formellement définie de la manière suivante : soit un ensemble d'**items** $\mathcal{I} = \{i_1, i_2, \ldots, i_n\}$ et soit $\mathcal{D}$ une base de données transactionnelle où chaque transaction contient un identifiant noté T_{id} et un ensemble non-vide d'items T tel que $T \in \mathcal{I}$. Un ensemble $(i_1, i_2, \ldots, i_k)$ est appelé **itemset**. On dit qu'une transaction T supporte l'itemset X si $X \subseteq T$. Le but de l'extraction des itemsets revient donc à énumérer tous les itemsets fréquents (i.e. chercher tous les itemsets qui sont supportés par plus de *minsupp* transactions dans $\mathcal{D}$).

ID Transactions	Items
T_1	$\{Cola, Pain\}$
T_2	$\{Chips, Pain\}$
T_3	$\{Pain, Yaourt\}$
T_4	$\{Chips, Pain, Chocolat\}$
T_5	$\{Chocolat, Cola, Chips, Yaourt, Pain\}$

Tab. 1.1: Une base de données exemple contenant 5 transactions

Exemple 1.1. Dans la base $\mathcal{D}$ de la table 1.1, en supposant que $minsupp = 3$, nous avons les itemsets fréquents suivants : $(pain)$, $(chips)$ et $(chips, pain)$.

L'extraction de ces itemsets permet de construire des connaissances basées sur des corrélations statistiques (telles que la *confiance*, le *lift* ou la *conviction*) appelées **règles d'associations** [AS94]. Une règle d'association est une règle du type $X \rightarrow Y$, où X est un itemset tel que $X \subseteq \mathcal{I}$, le conséquent Y est aussi un itemset tel que $Y \subseteq \mathcal{I}$ et $Y \cap X = \emptyset$.

Exemple 1.2. Une règle d'association possible dans la base $\mathcal{D}$ de la table 1.1 pourrait être $chips \rightarrow pain$.

Comme nous le disions précédemment, la problématique de l'extraction des motifs séquentiels est une extension de celle des itemsets dans laquelle nous prenons en considération la temporalité qui peut exister entre différents itemsets. En traduisant cela dans le problème du *panier de la ménagère*, nous pouvons extraire alors des corrélations sur les achats des clients sur *plusieurs*

Client	Date	Items
C_1	01/04/2008	{Pain,Cola}
C_1	02/04/2008	{Chips,Pain}
C_1	04/04/2008	{Pain}
C_1	18/04/2008	{Pain,Yaourt}
C_2	11/04/2008	{Chips}
C_2	12/04/2008	{Chocolat}
C_2	29/04/2008	{Yaourt, Pain, Chocolat}
C_3	05/04/2008	{Chips,Pain}
C_3	12/04/2008	{Yaourt,Pain}
C_4	06/04/2008	{Chips}
C_4	07/04/2008	{Chips}
C_4	08/04/2008	{Yaourt}

Fig. 1.2: Une base de données exemple contenant 4 clients (4 séquences de transactions)

transactions. Nous définissons formellement la problématique de l'extraction de motifs dans le chapitre mais intuitivement, si nous considérons la base de données illustrée dans la Table 1.2, nous recherchons à extraire des séquences du type : $\langle(chips)(pain, yaourt)\rangle$ que l'on peut traduire par *X% des clients ont acheté des chips puis après ont acheté du pain et des yaourts*.

Malheureusement, la recherche des motifs séquentiels est généralement considérée comme plus difficile que l'extraction des itemsets. Ceci est dû à l'espace de recherche exponentiel que le problème peut engendrer et qui est souvent inévitable lorsque les bases de données sont très denses. Pour s'en convaincre, supposons que la séquence $\langle(i_1)(i_2)\dots(i_{64})\rangle$ soit fréquente et que $\forall k \neq l, i_k \neq i_l$, alors cette séquence engendre $2^{64} - 1$ sous-séquence fréquente [TYH05].

2 Les flots de données

Avec la démocratisation des échanges commerciaux numériques et le passage des systèmes industriels vers des systèmes organisationnels dits de *réseaux d'échanges ouverts*, le volume de données générées à travers le monde a atteint une masse critique. Bien que cette augmentation de volume des données se soit accompagnée d'une évolution matérielle et logicielle, il devient de plus en plus coûteux de stocker et d'extraire des connaissances de qualités dans les immenses volumes actuels. Il apparaît alors que les techniques classiques de fouille de données ne sont plus adaptées pour de nouvelles applications telles que l'analyse des transactions boursières [Zhu], l'audit de navigation Web [GKMS01] ou l'analyse de logs de télécommunications [CFPR00]. En

effet, une nouvelle problématique, couplée à un nouveau modèle de données, a fait son apparition : les flots de données[2]. Dans ce modèle, les données manipulées sont des séquences d'informations obtenues généralement en temps réel, de manière continue et ordonnée. Ainsi, nous nous trouvons confrontés à des flots très importants de données (pour s'en convaincre, il suffit de compter le nombre de paquets TCP/IP transitant sur un réseau domestique en une journée) et l'accès rapide ainsi que le stockage de l'intégralité des données deviennent impossibles.

Bien que tous les flots possèdent les mêmes caractéristiques fondamentales de volume et de rapidité d'apparitions des informations, ils ne sont généralement pas tous du même type. Ainsi, un flot de données basé sur les transactions financières et les indices boursiers n'a pas le même volume qu'un flot basé sur un réseau informatique ou sur des données de capteurs. En outre, les besoins et les attentes sont différents. Pour ces raisons deux grands domaines de recherches se sont développés. Le premier domaine s'oriente vers la construction et l'analyse des systèmes de gestion de flux de données[3] [ABB+03, AAB+05, ACc+03]. Ces systèmes peuvent être vus comme une extension des différents axes de recherches qui ont été motivés dans le domaine des systèmes de bases de données classiques : la gestion, l'indexation et l'optimisation des requêtes, le traitement des jointures, les stockages etc...Le second domaine est celui de la fouille de données sur les flots afin d'extraire de la connaissance. Cependant les nouvelles contraintes (e.g. un seul accès possible aux données, résultats en temps réel) nécessitent de reconsidérer les approches traditionnelles d'extractions qui ne sont plus du tout adaptées.

Le thème de ce travail se situe à un carrefour entre le domaine de la fouille de données sur les flots et l'extraction de motifs séquentiels car en dépit de sa forte adéquation avec la nature des flots (arrivée en séquence des données à traiter), très peu de propositions fondées sur les motifs séquentiels n'existent à l'heure actuelle.

3 Principales contributions

Le travail de cette thèse présente différentes contributions à la problématique de la fouille de motifs séquentiels tant dans le contexte des flots de données que dans celui des bases de données statiques (voir Figure 1.3).

Un des points de départ de cette réflexion fut un ensemble de questions : *pourquoi n'existe-t-il pas plus de représentations condensées pour les motifs séquentiels ?* En effet, afin d'optimiser l'extraction de motifs (que ce soit des itemsets, des séquences, des arbres ou même des graphes), les chercheurs en fouille de données ont, dans la dernière décennie, travaillé sur l'extraction d'en-

[2] *Data streams* en anglais.

[3] Data Stream Management Systems

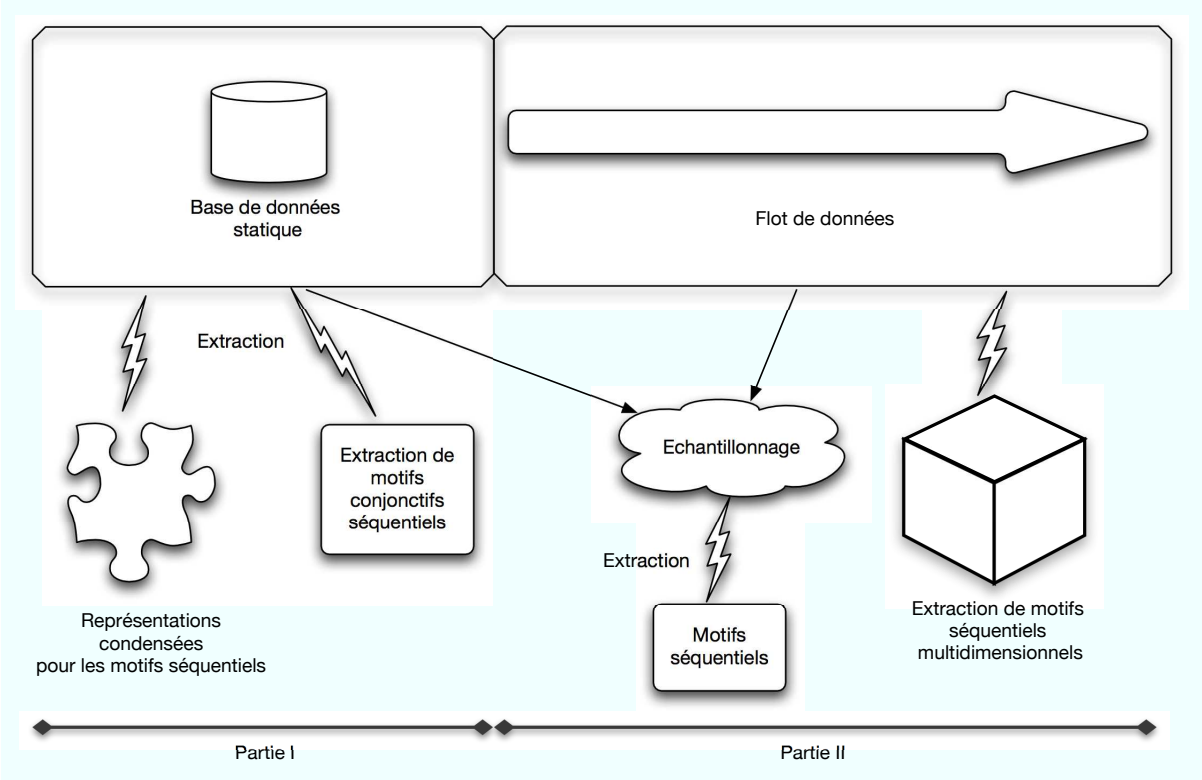


Fig. 1.3: Les différentes approches développées dans le cadre de cette thèse.

sembles de motifs de cardinalités plus petites mais qui contiennent la même puissance d'expression que l'ensemble original des motifs fréquents. Ces ensembles sont généralement appelés *représentations condensées* [MT96]. Ainsi, dans le cadre de l'extraction des itemsets, il existe de nombreuses représentations condensées telles que les représentations closes [PBTL99], les représentations par itemsets maximaux [BCF+05, GZ01], les représentations par itemsets non-dérivables [CG07], les représentations par ensembles libres [BBR03] etc... Pour ce qui est des motifs séquentiels, il n'existe que la représentation close [YHA03, WH04] et la représentation maximale [SA96]. *Pourquoi une telle différence ?*

Ensuite, *de quoi avons-nous réellement besoin pour réussir un échantillonnage des données avant une extraction de motifs séquentiels ?* La plupart des algorithmes d'extraction actuels ne peuvent pas gérer des bases avec un gros volume de données comme c'est le cas dans les bases de données réelles.*Comment pouvons-nous échantillonner ces bases de données pour permettre aux algorithmes de fonctionner de manière optimale sans nécessiter de gros espaces mémoires ?*

Et *si les flots de données sont multidimensionnels comment pouvons-nous extraire de la connaissance sous forme de séquences ?*, en effet, de jour en jour, les données deviennent de plus en plus complexes, plus difficiles à exprimer sur une seule dimension. *Comment pouvons-nous continuer à extraire de la connaissance sous forme de séquences tout en prenant compte de cette multidimensionnalité ?*

Nos travaux tout au long de ce mémoire visent à répondre à ces trois grandes questions. Nous avons donc suivi trois types d'approches :

1. Essayer de trouver des représentations condensées pour l'extraction de motifs qui seraient applicables dans le cas des bases de données statiques et des flots de données. Ce travail a permis de répondre à plusieurs problèmes théoriques liés à l'extraction des motifs séquentiels dans les bases de données statiques. De plus, nous présentons à la suite de ces résultats un nouveau type de motifs à extraire qui permettent l'extraction de *règles d'associations séquentielles*.
2. Essayer des trouver une approche d'échantillonnage en pré-traitement afin de *réduire le nombre d'entrées-sorties* nécessaires pour tous les algorithmes d'extraction de motifs séquentiels sur les bases de données statiques. Puis, essayer d'étendre cette approche aux flots.
3. Travailler sur des modèles de flots multidimensionnels et extraire des motifs séquentiels multidimensionnels sur les flots.

Les contributions associées à ce manuscrit sont :

– **Une nouvelle définition de relation d'équivalence pour les séquences** : Nous proposons dans la première partie de ce manuscrit une nouvelle définition de relation d'équivalence et de classe d'équivalence basées sur la notion de support d'une séquence. Ces nouvelles définitions permettent de décrire les différentes relations de supports ou de fréquences entre les motifs.

– **Une analyse théorique des représentations condensées possibles ou impossibles pour les motifs séquentiels** : Nous étudions dans la première partie, les possibilités de représentations condensées dans le cadre de l'extraction de motifs séquentiels. Nous montrons, généralement grâce aux classes d'équivalences, qu'il est impossible de construire des représentations condensées pour les motifs basées sur les représentations condensées pour les itemsets telles que la non-dérivabilité ou les ensembles libres [CG07, BBR03].

– **Un nouveau type de motifs à extraire** : *Les motifs conjonctifs séquentiels*, ces motifs sont basés sur les séquences et permettent d'extraire des règles d'associations séquentielles basées sur les mesures statistiques habituelles telles que la confiance, le lift ou la conviction.

– **Une nouvelle méthode pour calculer des bornes sur le support des motifs séquentiels** : Nous présentons une nouvelle méthode qui permet de calculer des bornes sur le support ou la fréquence des motifs séquentiels *sans avoir à passer par une étape de comptage*

sur la base de données.

– **Des méthodes d'échantillonnages pour les motifs séquentiels pour les bases de données statiques et les flots de données** : Ces méthodes se basent sur l'échantillonnage par réservoir. Nous introduisons des garanties sur la précision de l'échantillon et sa complétude ainsi qu'un algorithme de pré-traitement du flot dans le cadre de l'extraction de motifs séquentiels sur les flots.

– **Un algorithme d'extraction de motifs séquentiels multidimensionnels sur les flots** : Nous proposons un algorithme d'extraction de motifs séquentiels multidimensionnels sur les flots de données. L'espace de recherche associé à cette problématique étant très vaste, nous proposons un algorithme permettant un parcours efficace de cet espace de recherche et permettant d'extraire de la connaissance en temps *quasi-réel*.

Tout au long de ce mémoire nous utiliserons la base de données exemple suivante (voir Table 1.2) afin d'illustrer les différentes propositions. Cette base sera complétée dans la partie Flot de Données pour pouvoir prendre en compte d'une part le fait que les données sont disponibles dynamiquement mais également qu'elles peuvent avoir plusieurs dimensions.

Id. Séquence	Séquences
1	$(a, b)(a, b)(b, d, e)$
2	$(a, b, c, d, e)(b, e)$
3	(b, c, e)
4	$(a, c)(b, c, e)$
5	$(c)(c)(d)(b, c, e)$

Tab. 1.2: La base de données transactionnelles exemple

4 Organisation du mémoire

Le mémoire est organisé de la manière suivante :

Nous revenons sur la problématique de l'extraction de motifs séquentiels dans les bases de données statiques en introduisant les différentes définitions et en proposant un aperçu des travaux existants dans le chapitre 2.

La partie I du mémoire s'intéresse aux représentations condensées pour les motifs séquentiels. Dans le chapitre 3, nous analysons les différentes approches de représentations condensées existantes et nous étudions dans le chapitre 4 les possibilités de nouvelles représentations condensées pour les séquences. Les résultats théoriques issus de cette partie nous permettent d'introduire dans le chapitre 5 une nouvelle problématique d'extraction de motifs qui permet la génération de *règles d'associations séquentielles*.

La partie II du mémoire s'intéresse à la prise en compte des données sous la forme de flots. Nous introduisons la problématique des flots de données ainsi que les différentes méthodes de fouilles de données associées à ce nouveau modèle dans le chapitre 6. Puis, dans le chapitre 7, nous proposons une nouvelle approche d'extraction basée sur une technique d'échantillonnage qui garantit la précision et la complétude des résultats d'extraction de motifs séquentiels. Enfin dans le chapitre 8 nous nous intéressons à la problématique des flots de données multidimensionnelles et proposons une nouvelle approche pour extraire des motifs séquentiels multidimensionnels sur des flots.

Enfin dans le chapitre 9, nous concluons et proposons un bilan et les différentes perspectives de recherche associées.

Chapitre 2

Motifs séquentiels : contexte et problématique

L'objectif de ce chapitre est de présenter la problématique de l'extraction de motifs séquentiels au sens large. Dans la section 1, nous introduisons les principales définitions. Nous revenons également sur l'exemple présenté dans le chapitre 1 afin d'illustrer la problématique. La section 2 présente les différents travaux de recherche existants. Ce chapitre se termine par une discussion.

1 Définitions et problématique

Initialement introduite dans [Sri95], la notion de séquence est définie de la manière suivante.

Définition 2.1 (Base de données transactionnelles)**.** Une base de données transactionelles $\mathcal{D}$ basée sur les items de $\mathcal{I}$ est un ensemble fini de paires (SID, T), appelées transactions, avec $SID \in \{1, 2, \ldots\}$ un identifiant et $T \in \mathcal{T}(\mathcal{I})$ une séquence construite sur $\mathcal{I}$. Pour deux transactions quelconques $(SID_1, T_1) \neq (SID_2, T_2) \in \mathcal{D}$ implique que $SID_1 \neq SID_2$.

Définition 2.2 (Itemset)**.** Un itemset est un ensemble non vide d'items noté $(i_1 i_2 ... i_k)$.

Définition 2.3 (Séquence)**.** Une séquence est une liste ordonnée, non vide, d'itemsets notée $\langle (it_1) \ldots (it_n) \rangle$ où (it_j) est un itemset. $\mathcal{T}(\mathcal{I})$ représente l'ensemble (infini) de toutes les séquences possibles à partir des items présents dans l'ensemble $\mathcal{I}$.

Exemple 2.1 (Sémantique)**.** Le problème de l'extraction de motifs séquentiels est souvent relié au problème du *panier de la ménagère*. Soit C un client et $S{=}\langle (a)(b, c)(d) \rangle$, la séquence de données

représentant les différents achats de ce client. S peut être interprétée par *"C a acheté l'item a, puis en même temps les items b et c et enfin l'item d"*. Dans ce cas, la séquence S contient 3 itemsets et la taille de cette séquence est 4.

Définition 2.4 (Inclusion)**.** Une séquence $S' = \langle(it'_1)\dots(it'_n)\rangle$ est une sous-séquence de $S = \langle(it_1)\dots(it_m)\rangle$, notée $S' \preceq S$, si $\exists\ i_1 < i_2 < \dots\ i_j\ \dots < i_n$ tels que $it'_1 \subseteq it_{i_1}$, $it'_2 \subseteq it_{i_2}$, $\dots it'_n \subseteq it_{i_n}$. Si $S \not\preceq S'$ et $S' \not\preceq S$, les séquences sont dites incomparables et sont notées $S \prec\succ S'$. De plus, une séquence est dite régulière si chaque itemset it_j contient le même unique item i.

Exemple 2.2.

La séquence $S' = \langle(a)(b,c)(d)\rangle$ est incluse dans la séquence $S = \langle(a,d,e)(g,h)(f)(b,c,e)(d,e,f)\rangle$ (i.e. $S' \preceq S$) car $(a) \subseteq (a,d,e)$, $(b,c) \subseteq (b,c,e)$ et $(d) \subseteq (d,e,f)$. En revanche, $\langle(a)(b)\rangle \not\preceq \langle(a,b)\rangle$ (et vice versa). Les deux séquences $\langle(a)(b)\rangle$ et $\langle(a,b)\rangle$ sont dites incomparables.

Définition 2.5 (Support)**.** Le support d'une séquence S dans une base de données transactionnelles $\mathcal{D}$, noté $Support(S,\mathcal{D})$ ou $Support(S)$ quand le contexte est clair, est défini tel que :

$$Support(S,\mathcal{D}) = |\{(SID,T) \in \mathcal{D} | S \preceq T\}| \tag{2.1}$$

Définition 2.6 (Fréquence)**.** La fréquence de S dans $\mathcal{D}$, noté f_S, est définie telle que :

$$f_S = \frac{Support(S,\mathcal{D})}{|\mathcal{D}|} \tag{2.2}$$

Remarque 2.1. Une séquence de données n'est prise en compte qu'une seule fois pour calculer le support ou la fréquence d'une séquence, i.e. le processus d'extraction de séquences ne tient compte de l'apparition de la séquence qu'une seule fois.

Avec ces définitions, nous pouvons maintenant décrire formellement le problème d'extraction des motifs séquentiels et sa solution.

Définition 2.7 (Extraction des motifs séquentiels fréquents)**.** Soit $\mathcal{D}$ une base de données transactionnelles. Étant donné un seuil de support minimal (ou un seuil de fréquence minimale) σ, le problème d'extraction de motifs séquentiels est l'énumération de toutes les séquences S dans $\mathcal{D}$

telles que $f_S \geqslant \sigma$. L'ensemble des motifs séquentiels pour la valeur σ dans la base de données $\mathcal{D}$ est noté $FSeqs(\mathcal{D}, \sigma)$.

$$FSeqs(\mathcal{D}, \sigma) := \{S \mid Support(S, \mathcal{D}) \geqslant \sigma\} \tag{2.3}$$

Avec $0 < \sigma \leqslant 1$ dans le cas où σ est un seuil de fréquence et $0 < \sigma \leqslant |\mathcal{D}|$ dans le cas où σ est un seuil de support.

La propriété suivante est considérée comme la propriété centrale pour la construction d'algorithmes efficaces d'extractions de motifs.

Propriété 1 (Antimonotonie [AIS93])**.** Soit S' et S deux séquences. Si $S' \preceq S$ alors $Support(S') \geqslant Support(S)$ (ou $f_{S'} \geqslant f_S$).

La propriété suivante est une conséquence de la propriété 1.

Propriété 2. Soit S' une séquence non fréquente. Quelle que soit S telle que $S' \preceq S$, S est une séquence non fréquente.

En effet, d'après cette propriété, $Support(B) \leqslant Support(A) < \sigma$, donc B n'est pas fréquent.

Id. Séquence	Séquences
1	$(a, b)(a, b)(b, d, e)$
2	$(a, b, c, d, e)(b, e)$
3	(b, c, e)
4	$(a, c)(b, c, e)$
5	$(c)(c)(d)(b, c, e)$

Tab. 2.1: La base de données transactionnelles exemple

Exemple 2.3. Considérons la base de données $\mathcal{D}$ du chapitre 1 et représentée dans la table 2.1. Avec un seuil de fréquence minimum σ de $\frac{3}{5}$ soit 60% (i.e. pour qu'une séquence s soit retenue, il faut qu'au moins trois séquences dans la base de données la supportent), les séquences fréquentes sont alors les suivantes :

$$FSeqs(\mathcal{D}, \sigma) = \left\{ \begin{array}{lll} \langle a \rangle : 3 & \langle (a)(b) \rangle : 3 & \langle (c)(b, e) \rangle : 3 \\ \langle b \rangle : 5 & \langle (a)(e) \rangle : 3 & \langle (a)(b, e) \rangle : 3 \\ \langle c \rangle : 4 & \langle (b, c) \rangle : 4 & \\ \langle d \rangle : 3 & \langle (b, e) \rangle : 5 & \\ \langle e \rangle : 5 & \langle (c)(b) \rangle : 3 & \\ & \langle (c, e) \rangle : 4 & \\ & \langle (c)(e) \rangle : 3 & \end{array} \right\}$$

2 Travaux antérieurs

Depuis sa définition par R.Agrawal et R.Srikant en 1993 [AIS93], de nombreuses approches ont été proposées ces dernières années afin d'améliorer les méthodes d'extractions de motifs séquentiels [Sri95, SA96, AFGY02, HPMa+00, KPWD03, MCP98, PHMa+01, PHMa+04, Zak01]. De manière générale, l'extraction de motifs séquentiels peut être considérée comme un problème d'énumération de séquences respectant une contrainte de fréquence (ou de support). Comme nous l'avons vu en introduction, cette problématique est assez proche de celle de la recherche des itemsets afin d'extraire des règles d'association où la taille de l'espace de recherche correspond à $2^{|\mathcal{I}|}$ où $|\mathcal{I}|$ correspond au nombre d'items différents présents dans une base de données $\mathcal{D}$. Pour les séquences, la prise en compte de la relation d'ordre entre les itemsets implique un espace de recherche encore plus grand que celui de la problématique de recherche d'itemsets. Pour s'en convaincre, prenons l'exemple où nous considérons une séquence $\langle (it_1) \dots (it_m) \rangle$ et supposons que $n_i = |it_i|$ représente la cardinalité d'un itemset alors l'espace de recherche, i.e. l'ensemble de toutes les séquences potentielles, est $2^{n_1+n_2+\dots n_m}$.

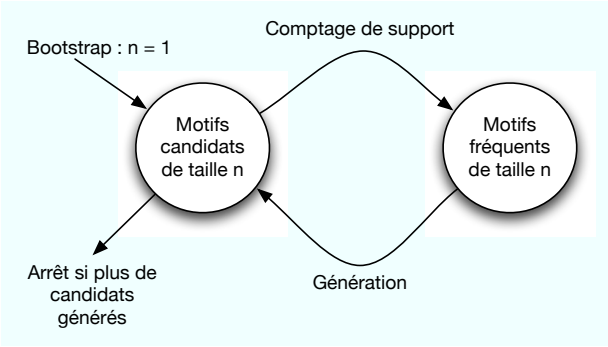


Fig. 2.1: Diagramme représentant la méthode *générer-élaguer*

Une méthode naïve d'extractions de motifs (tant pour les itemsets que pour les séquences) serait de générer tous les motifs possibles et de compter leurs supports dans la base de données. Malheureusement cette approche ne peut pas être traitée en un *temps raisonnable* puisque si l'on suppose que l'ensemble des items $|\mathcal{I}|$ contient plus de 263 items, alors il faudra générer plus de motifs que le nombre d'atomes dans l'univers ($2^{263} > 10^{79}$). Pour résoudre ce problème de parcours de l'espace de recherche, les premières approches d'extractions de motifs (tant d'itemsets que de séquences) ont adopté une méthode dite de **générer-élaguer** en se basant sur la propriété 1 qui stipule que le support (ou la fréquence) décroît de manière monotone lors de l'extension du motif (souvent appelé le paradigme *Apriori*). Ainsi, si un motif est considéré non fréquent, il n'est pas nécessaire de générer ses super-motifs puisqu'ils seront aussi non fréquents. Cette méthode consiste donc à générer un ensemble de *motifs candidats* puis une fois la génération finie, à compter le sup-

port de chaque élément dans l'ensemble. Les éléments qui seront non fréquents seront éliminés de l'ensemble (élagués). Les éléments restants sont ainsi réutilisés dans une étape de génération qui doit être très optimisée afin de générer le plus petit ensemble possible de candidats. Cette méthode s'arrête lorsqu'il n'y a plus de candidats fréquents. La figure 2.1 illustre le fonctionnement de la méthode générer-élaguer et l'algorithme 1 présente un algorithme générique d'extraction de motifs.

Algorithme 1 : Algorithme générique pour l'extraction de motifs

Data : Base de données $\mathcal{D}$, un seuil de fréquence σ

Result : L'ensemble des motifs fréquents $FSeqs(\mathcal{D}, \sigma)$

```
1  begin
2      C_1 ← {{i} | i ∈ I};
3      k ← 1;
4      FSeqs(D, σ) ← ∅;
5      while C_k ≠ ∅ do
6          Compter les fréquences des motifs de l'ensemble C_k dans la base D;
7          FSeqs(D, σ) ← FSeqs(D, σ) ∪ {les motifs fréquents présents dans C_k};
8          C_{k+1} ← Générer ensemble contenant nouveaux candidats à partir de C_k;
9          k ← k + 1;
10 end
```

Dans le cadre de l'extraction de motifs séquentiels, l'étape de génération est plus complexe que celle pour les itemsets. Ainsi, dans le cas des séquences, il existe deux manières d'étendre une séquence par un item : l'extension de séquence (*S-Extension*) et l'extension d'itemset (*I-Extension*). Dans une *S-Extension*, l'item est ajouté à la séquence comme nouvel itemset. Dans le cas de la *I-Extension*, l'item est ajouté au dernier itemset de la séquence à étendre.

Exemple 2.4. Soit $s = \langle (a)(b) \rangle$ une séquence, une I-Extension de la séquence avec l'item c donne la séquence s' suivante : $s' = \langle (a)(b, c) \rangle$. Une *S-Extension* de la séquence avec l'item c donne la séquence s'' suivante : $s'' = \langle (a)(b)(c) \rangle$.

Les méthodes existantes différent essentiellement sur la manière de parcourir l'espace de recherche (largeur d'abord ou profondeur d'abord) et sur les structures de données utilisées pour *indexer* la base de données et faciliter une énumération rapide. Les algorithmes d'extraction de motifs séquentiels peuvent être classés en trois grandes catégories :

1. Méthodes horizontales
2. Méthodes verticales
3. Méthodes par projection

Nous décrivons dans les sections suivantes ces trois catégories ainsi que les principaux algorithmes associés.

2.1 Méthodes horizontales

Algorithme pionnier : GSP et sa structure

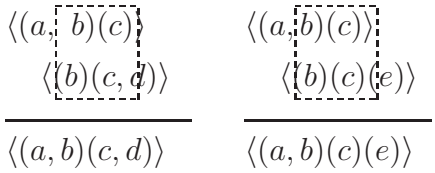


FIG. 2.2: Extension d'une séquence : I-extension (gauche) et S-Extension (droite)

La méthode GSP (*Generalized Sequential Patterns*) [SA96] a été l'une des premières propositions pour résoudre la problématique des motifs séquentiels (ce travail fait suite à [Sri95]). Les auteurs, en définissant la problématique de l'extraction de motifs séquentiels, ont également proposé un algorithme reprenant les principes de l'algorithme *Apriori*, conçu pour l'extraction d'itemsets fréquents. Cependant, les difficultés relatives à la prise en compte de la temporalité ont rapidement conduit à la mise en place d'une méthode de génération de candidats adaptée à ce contexte. Celle-ci maintient cependant les principes d'une recherche *en largeur d'abord* puisque les candidats sont générés en fonction de leur longueur et non de leur préfixe.

GSP est un algorithme basé sur la méthode générer-élaguer afin de minimiser le nombre de passes sur la base de données. La technique généralement utilisée par les algorithmes de recherche de séquences est basée sur une création de candidats (par I-extension et S-extension), suivie du test de ces candidats pour confirmer leur fréquence dans la base. La génération de candidats se fait dans le cadre général par auto-jointure sur le dernier ensemble extrait de motifs fréquents. Ainsi, pour générer les séquences candidates de taille k, l'algorithme GSP fait une opération de jointure sur l'ensemble de motifs fréquents de taille $k-1$. Nous illustrons cette opération de génération dans la figure 2.2 en étendant la séquence $\langle(a,b)(c)\rangle$. Pourtant, même en bénéficiant de propriétés relatives aux séquences et à leur fréquence d'apparition, ces techniques sont tout de même contraintes de compter la fréquence des séquences avant de les déterminer fréquentes (ou non).

Pour évaluer le support de chaque candidat en fonction d'une séquence de données, GSP utilise une structure d'arbre de hachage (*hash-tree*) destinée à organiser les candidats. Les candidats sont stockés en fonction de leur préfixe. Pour ajouter un candidat dans l'arbre des séquences candidates, GSP parcourt ce candidat et effectue la descente correspondante dans l'arbre. Pour

trouver quelles séquences candidates sont incluses dans une séquence de données, GSP parcourt l'arbre en appliquant une fonction de hachage sur chaque item de la séquence de données. Quand une feuille est atteinte, elle contient des candidats potentiels pour la séquence de données.

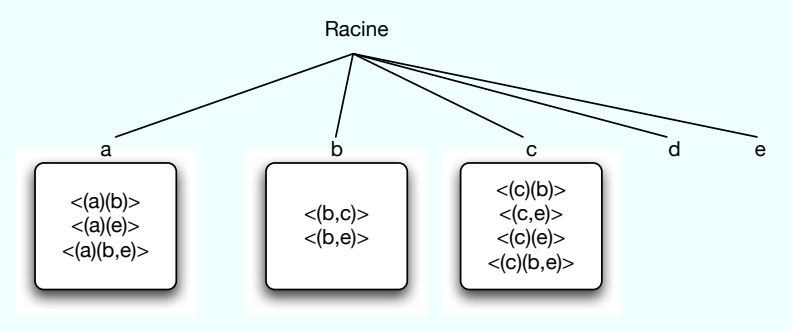


Fig. 2.3: La structure de données utilisée par l'algorithme GSP

Exemple 2.5. La figure 2.3 montre la structure de données de l'algorithme GSP à la fin de l'extraction de motifs séquentiels sur la base de données $\mathcal{D}$ du chapitre 1 et représentée dans la table 2.1. Comme on peut le remarquer, une feuille dans l'arbre peut contenir plusieurs séquence. Ainsi, quand GSP atteint cette feuille, il n'a aucun moyen de savoir quelle séquence l'a conduit jusqu'à cette feuille. Pour le savoir, GSP doit tester les séquences contenues dans la feuille afin de savoir quelle séquence doit avoir sa fréquence incrémentée.

Une première amélioration : PSP et sa structure d'arbre prefixé

La structure de données de l'algorithme GSP a été améliorée dans PSP (*Prefix Tree for Sequential Pattern*) [MCP98] où les auteurs proposent d'utiliser une structure d'arbre préfixé.

Le principe de cette structure consiste à factoriser les séquences candidates en fonction de leur préfixe. De plus, pour prendre en compte le changement d'itemset, l'arbre est doté de deux types de branches. Le premier type, entre deux items, signifie que les items sont dans le même itemset alors que le second signifie qu'il y a un changement d'itemset entre ces deux items. Ainsi, l'arbre de préfixes ne stocke plus les candidats dans les feuilles mais permet de retrouver les candidats de la façon suivante : tout chemin de la racine à une feuille représente un candidat et tout candidat est représenté par un chemin de la racine à une feuille. Cette nouvelle structure de données permet ainsi une nette amélioration des performances lors de l'extraction de motifs séquentiels.

Exemple 2.6. La figure 2.3 montre la structure de données de l'algorithme PSP à la fin de l'extraction de motifs séquentiels sur base de données $\mathcal{D}$ du chapitre 1 et représentée dans la table 2.1.

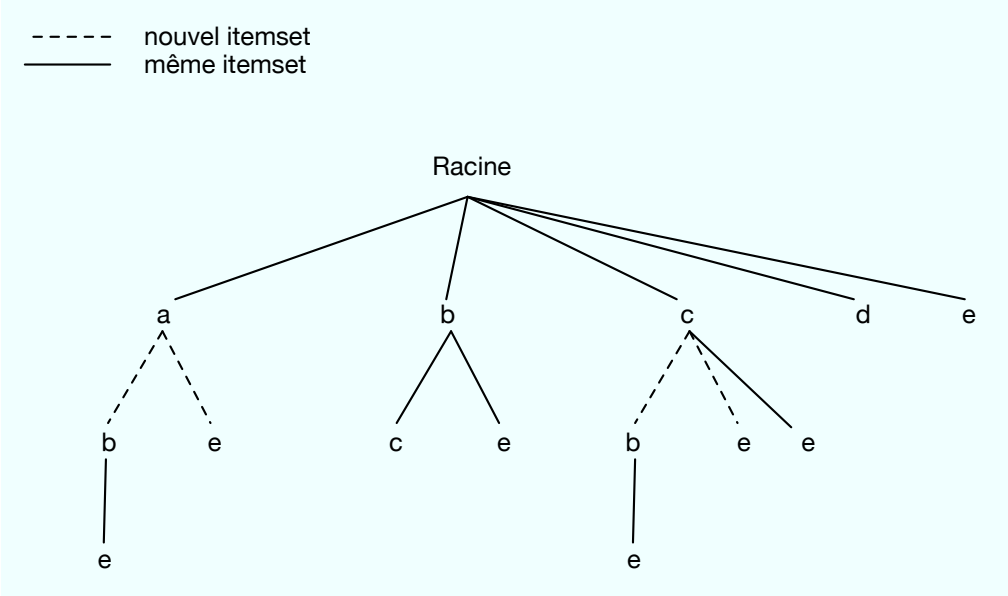


Fig. 2.4: La structure de données utilisée par l'algorithme PSP

SPAM : une représentation binaire des données

Ayres et al. ont proposé l'algorithme SPAM [AFGY02]. Cet algorithme se démarque des autres approches par l'utilisation d'une représentation par vecteurs de bits de la base de données. Lors de la première passe sur la base de données, un vecteur binaire est construit pour chaque item. Dans ce vecteur chaque bit représente une séquence de la base, si l'item est présent dans la séquence, le bit correspondant est activé (valeur à 1). Cette représentation permet d'optimiser le processus de comptage de supports puisque cette opération revient en fait à utiliser la fonction booléenne AND. Cependant, l'implémentation publique de cet algorithme[1] souffre de certaines limitations qui l'empêche d'être appliqué sur des bases de données réelles.

2.2 Méthodes verticales

A partir des années 1999 et 2000, les chercheurs dans le domaine de la fouille de motifs ont commencé à s'intéresser de plus en plus à l'optimisation de l'extraction. C'est dans ce contexte qu'est apparu l'algorithme SPADE [Zak01].

SPADE : Une représentation inversée et des classes d'équivalences

L'algorithme SPADE est le premier algorithme à proposer et utiliser une représentation verticale de la base de données pour extraire de manière plus efficace des motifs séquentiels. Cette

[1] http ://himalaya-tools.sourceforge.net/Spam/

approche est en fait une extension de l'algorithme CHARM [ZH02] qui permet d'extraire des itemsets fréquents. Une représentation en base de données verticales consiste à inverser la méthode d'indexation de la base de données. De plus, l'algorithme SPADE introduit une nouvelle représentation des motifs séquentiels en les regroupant selon des classes d'équivalences construites sur une relation d'équivalence prenant en compte le préfixe des séquence. Ainsi, ces classes d'équivalences permettent de décomposer le problème initial en sous-problèmes plus facile à gérer en mémoire.

Les classes d'équivalences de l'algorithme SPADE sont définies à partir d'une relation d'équivalence sur le préfixe. Ainsi, deux séquences de taille k sont dans la même classe d'équivalence si elles ont un préfixe commun de taille $k-1$. Plus formellement, une classe d'équivalence est définie de la manière suivante :

$$[\rho \in FSeqs(\mathcal{D},\sigma)_{k-1}] = \{s \in FSeqs(\mathcal{D},\sigma)_k | Pref_{k-1}(s) = \rho\} \tag{2.4}$$

Où $Pref_{k-1}(s)$ représente la sous-séquence de taille $k-1$ qui est préfixe de la séquence s et $FSeqs(\mathcal{D},\sigma)_k$ représente les séquences fréquentes de taille k. La génération de candidats se fait par opération de jointures entre les différents éléments des classes d'équivalences. Quand au comptage de support, il est effectué en calculant les intersections possibles dans la base de données inversée.

Exemple 2.1. La figure 2.5 représente la base de données exemple dans le format vertical pour les séquences $\langle a \rangle$, $\langle b \rangle$ et $\langle (a)(b) \rangle$. En effet, la transformation consiste à associer à chaque k-séquence l'ensemble des couples (Id Séquence, transaction) qui lui correspondent dans la base. Le support d'une séquence est le cardinal de l'ensemble constitué par les identifiants de séquences. Dans le cas de la séquence $\langle a \rangle$, l'ensemble des identifiants est : $\{1, 2, 4\}$ de cardinal 3, le support de cette séquence est donc 3.

Séquence $\langle a \rangle$	
Id. Séq.	Transactions
1	1
1	2
2	1
4	1

Séquence $\langle b \rangle$	
Id. Séq.	Transactions
1	1
1	2
1	3
2	1
2	2
3	1
4	2
5	4

Séquence $\langle (a)(b) \rangle$	
Id. Séq.	Transactions
1	1
1	2
2	1
2	2
4	1

Fig. 2.5: La base de données au format vertical pour les séquences $\langle a \rangle$, $\langle b \rangle$ et $\langle (a)(b) \rangle$

2.3 Méthodes par projection

Dans [HPMa+00], les auteurs proposent l'algorithme FreeSpan (*Frequent pattern projected Sequential pattern mining*). L'idée générale est d'utiliser des projections récursives de la base de données en fonction des items fréquents. La base est projetée en plusieurs sous-bases, par conséquent, les temps de réponses sont améliorés car chaque base projetée est plus petite et facile à traiter. Ce travail est le point de départ d'autres études sur la projection de bases de données en recherche de motifs séquentiels. FreeSpan présente tout de même un défaut selon ses auteurs : une sous-séquence peut être générée par n'importe quelle combinaison dans une séquence, l'algorithme doit donc conserver la totalité de la séquence dans la base d'origine sans réduire sa taille.

L'algorithme PrefixSpan

La méthode PrefixSpan, présentée dans [PHMa+01], se base elle aussi sur l'idée de projection de bases de données de manière récursive. L'objectif premier des auteurs est de réduire le nombre de candidats générés. Pour parvenir à cet objectif, PrefixSpan propose d'analyser les préfixes communs que présentent les séquences de données de la base à traiter. A partir de cette analyse, l'algorithme construit des bases de données intermédiaires qui sont des projections de la base d'origine déduites à partir des préfixes identifiés (i.e. elles ne contiennent que les postfixes). Ensuite, dans chaque base obtenue, PrefixSpan applique un comptage du support des différents items afin de faire croître la taille des motifs séquentiels découverts.

Trois sortes de projections sont alors mises en place pour réaliser cette méthode : la projection dite *"niveau par niveau"*, la *"bi-projection"* et la *"pseudo-projection"* qui est une méthode d'indexation permettant de considérer plusieurs bases virtuelles à partir d'une seule, dans le cas où les bases générées ne pourraient être maintenues en mémoire en raison de leurs tailles.

La première étape d'extraction de motifs dans l'approche PrefixSpan consiste à compter le support des séquences de taille 1 (i.e. les items). Ensuite, la base de données est divisée en plusieurs partitions selon le nombre d'items fréquents. Chaque partition est en fait une base de données qui prend un des items fréquents comme préfixe. En procédant à un comptage de tous les items fréquents sur les bases de données projetées, PrefixSpan extrait ainsi toutes les séquences fréquentes de taille 2. Ce processus est répété récursivement jusqu'à ce que les bases de données projetées soient vides, ou qu'il n'y ait plus de séquences fréquentes possibles. L'exemple 2.2 illustre le fonctionnement de l'algorithme sur notre base de données exemple.

Exemple 2.2. Nous illustrons dans la table 2.3 l'exécution de l'algorithme PrefixSpan sur la base de données exemple $\mathcal{D}$. La fréquence minimale est toujours de $\frac{3}{5}$. Premièrement, PrefixSpan

Id. Séquence	Séquences
1	$(a, b)(a, b)(b, d, e)$
2	$(a, b, c, d, e)(b, e)$
3	(b, c, e)
4	$(a, c)(b, c, e)$
5	$(c)(c)(d)(b, c, e)$

Tab. 2.2: La base de données transactionnelles exemple

trouve les séquences fréquentes de taille 1 (colonne Préfixe). Ensuite, l'algorithme divise l'espace de recherche en procédant aux projections de la base de données selon les préfixes possibles. Toutes les bases projetées à l'étape 2 sont représentées sous la colonne *Bases projetées*. Le recomptage de support des séquences de taille 1 dans ces bases projetées permet de compter le support des séquences. Les séquences fréquentes sont présentées dans la colonne *Motifs fréquents*.

Les algorithmes LAPIN et PAID

Plusieurs nouveaux algorithmes ont vu le jour à partir des travaux de J. Pei et J. Han [HPMa$^+$00, PHMa$^+$01] sur les méthodes par projections (*Pattern Growth*). Dans [YWK05], les auteurs proposent l'algorithme LAPIN afin d'extraire des séquences fréquentes. Cet algorithme se base sur l'idée simple que seule la dernière position d'un item α dans une séquence s permet de juger si une k-séquence fréquente peut être étendue avec l'item α pour former une $k+1$-séquence. Cette approche permet en fait de limiter encore plus l'espace de recherche parcouru lors de l'extraction de motifs puisque l'algorithme ne s'intéresse qu'à construire une table d'index des items pour chaque séquence dans la base de données. Une amélioration de l'algorithme LAPIN est proposé avec l'algorithme PAID [YKW06] qui s'appuie sur une optimisation du précédent algorithme LAPIN en utilisant une représentation verticale de la base de données.

3 Discussion

Un compromis nécessaire

Tout au long de cet examen des travaux antérieures, nous avons pu mettre en exergue la difficulté et la complexité de la tâche d'extraction des motifs et surtout des motifs séquentiels. Le principal problème sur lequel échouent généralement les algorithmes d'extractions est la génération de candidats. En effet, dans le meilleur des cas, un algorithme d'extraction devrait générer le moins de candidats possibles, ce qui reviendrait à générer uniquement les motifs qui seront fréquents.

Préfixe	Bases projetées	Motifs fréquents
$\langle a \rangle$	$\langle (\_, b)(a, b)(b, d, e) \rangle$ $\langle (\_, b, c, d, e)(b, e) \rangle$ $\langle (\_, c)(b, c, e) \rangle$	$\langle (a)(b) \rangle$, $\langle (a)(e) \rangle$,$\langle (a)(b, e) \rangle$
$\langle b \rangle$	$\langle (a, b)(b, d, e) \rangle$ $\langle (\_, c, d, e)(b, e) \rangle$ $\langle (\_, c, e) \rangle$ $\langle (\_, c, e) \rangle$ $\langle (\_, c, e) \rangle$	$\langle (b, c) \rangle$, $\langle (b, e) \rangle$
$\langle c \rangle$	$\langle (\_, d, e)(b, e) \rangle$ $\langle (b, c, e) \rangle$ $\langle (c)(d)(b, c, e) \rangle$	$\langle (c)(b) \rangle$, $\langle (c, e) \rangle$, $\langle (c)(e) \rangle$, $\langle (c)(b, e) \rangle$
$\langle d \rangle$	$\langle (\_, e) \rangle$ $\langle (\_, e)(b, e) \rangle$ $\langle (b, c, e) \rangle$	
$\langle e \rangle$	$\langle (b, e) \rangle$	

TAB. 2.3: Trace de l'exécution de PrefixSpan avec la base de données exemple

Or cet idéal n'est généralement pas du tout atteint. Les algorithmes sont donc amenés à essayer d'optimiser leurs générations de candidats en se basant sur l'approche *générer-élaguer* tout en tenant compte des besoins mémoires que peuvent engendrer les processus d'extractions.

Pour l'algorithme GSP, qui ne charge pas la base de données en mémoire, le facteur temps est un obstacle majeur lors de la fouille sur de grandes bases de données. D'autres algorithmes, tels que PrefixSpan et FreeSpan, sont basés sur le paradigme *pattern-growth* et proposent d'éviter la génération[1] de candidats en projetant la base de données en de multiples sous-bases selon certaines conditions. Ces algorithmes nécessitent généralement que la base de données puisse être chargée en mémoire centrale lors de l'opération de fouille. Si ce n'est pas possible, une méthode dite de *pseudo-projection* permet une sorte de *swapping* et de *cache* de la mémoire vers un espace mémoire non-volatile (disque) afin de permettre l'extraction de motifs. Un autre algorithme, SPADE, propose quant à lui la transformation et la réécriture de la base de données afin de pouvoir diviser le processus d'extraction en sous-problèmes de tailles plus petites qui peuvent être ainsi plus facilement traités en mémoire. Or, la reformulation même de la base de données de sa forme classique horizontale vers la forme verticale est déjà coûteuse en mémoire. Comme pour les approches basées sur les projections de bases de données, les auteurs dans [Zak01] proposent d'utiliser un stockage sur un disque dur afin d'éviter de surcharger la mémoire. Il est effectivement

[1] Cette absence de génération est discutée et même mise en doute dans le travail de thèse de B. Goethals [Goe02].

intéressant de remarquer que les auteurs des différentes approches ne proposent pas les résultats des expérimentations lorsqu'un système de *cache* disque est utilisé.

Ainsi, et globalement durant la décennie qui suivit l'introduction de la problématique d'extraction des motifs séquentiels, les chercheurs ont essayé de trouver un juste milieu entre la gestion mémoire de leurs algorithmes et l'efficacité de la génération de candidats et de comptage des supports pour les motifs. Mais dans les dernières années un ensemble de publications tend à orienter la recherche dans le domaine de l'extraction de motifs séquentiels vers de nouvelles approches qui ne sont plus nécessairement orientées vers l'optimisation de la structure de données d'un algorithme ou uniquement dans l'optimisation de la génération. Ces méthodes trouvent généralement leurs racines dans des approches plus théoriques et formelles telles que la *théorie des treillis, l'analyse de concepts formels, les méthodes statistiques etc....*

Vers de nouvelles contraintes

Comme nous l'avons déjà abordé dans l'introduction de ce mémoire, l'objectif des approches d'extraction de connaissances est de proposer à l'utilisateur d'extraire un sous ensemble de *pépites de connaissances* d'un gros volume de données. Cependant dans le cas des séquences (et également des itemsets), ce n'est pas toujours le cas. Ainsi, l'analyste, ou le décideur se retrouve généralement avec une *montagne de connaissances* encore plus difficile à appréhender et interpréter que ses données de départ. Pour répondre à cette nouvelle sous-problématique de l'extraction de motifs, les chercheurs ont développés le concept de *représentation condensée* [MT96]. Ces représentations sont généralement basées sur des acquis théorique des domaines de la théorie des treillis, de l'analyse de concepts formels et parfois de la branche des combinatoires en mathématiques. Ces représentations permettent de limiter les résultats de l'extraction en ne présentant que des connaissances concises, précises et souvent non-redondantes. Bien que de nouvelles représentations condensées soient très fréquemment présentées et découvertes dans le cadre de l'extraction d'itemsets, elles restent malheureusement très limitées dans le cadre de l'extraction de motifs séquentiels.

De plus, alors que des méthodes d'échantillonnages ont été présentées dans le cadre de l'extraction d'itemsets [Toi96, BCG03, CHHP05], elles ne sont que trop peu représentées dans le cadre de l'extraction de motifs séquentiels [LC05]. Nous pensons que les méthodes d'échantillonnage, ont une place à prendre dans le cadre de l'extraction de motifs séquentiels, d'autant plus, qu'avec l'augmentation des volumes de données réelles, des capacités de stockages et l'apparition des flots de données, il devient de plus en plus acceptable d'avoir des *résultats approximés* si l'on peut y associer des garanties sur la précision ou les taux d'erreurs.

Première partie

Extraction de Motifs Séquentiels dans les Bases de Données Statiques

Data Mining is the art of counting
Heikki Mannila (2000) —

Un problème inhérent à la fouille de motifs, limitant son application dans les bases de données réelles, est le gros volume de connaissances extraites. De plus, du point de vue de l'analyste ou du décideur, l'interprétation de ces connaissances est souvent fastidieuse car elles contiennent généralement beaucoup de redondances empêchant ainsi de trouver les *pépites de connaissances*.

C'est à partir de ces observations que des chercheurs se sont attelés à essayer de limiter le nombre de motifs extraits en développant le concept de *représentation condensées* [MT96]. Ces nouveaux ensembles de motifs ne sont rien d'autre qu'un sous-ensemble de l'ensemble des motifs fréquents mais contenant la même puissance d'expression que ce dernier. Ainsi, dans le cadre de la fouille d'itemsets, il existe une myriade de représentations possibles (il est même possible d'un point de vue théorique d'en imaginer une infinité grâce aux ensembles k-libres). Malheureusement, dans le cadre des motifs séquentiels, il n'en existe que deux : la représentation par motifs maximaux et celle par motifs clos. *Pourquoi une telle différence ?*

Nous répondrons à cette question essentielle dans cette partie. Ainsi, dans le chapitre 3 nous revenons sur la notion de représentation condensée et présentons les travaux antérieurs dans le cadre de l'extraction d'itemsets et de séquences. Le chapitre 4 étudie la possibilité d'application des concepts de *non-derivabilité* et *k-liberté* pour les séquences en étudiant la possibilité de dériver des règles sur la fréquence des motifs séquentiels. Enfin, le chapitre 5 présente un nouveau type de motifs : les *motifs conjonctifs séquentiels*. Ce motif est construit à partir des motifs séquentiels et possède des propriétés intéressantes pour les représentations condensées ainsi que pour l'extraction de règles d'association dites *séquentielle*.

Chapitre 3

Représentations condensées pour les motifs séquentiels

1 Introduction

L'objectif de ce chapitre est de présenter la problématique des représentations condensées dans le cadre de l'extraction d'itemsets et de motifs séquentiels. Dans ce chapitre, nous dressons un panorama des différentes approches existantes. Outre le caractère informatif de ce chapitre, nous nous focaliserons sur les différences et les besoins qui existent pour les représentations condensées pour les itemsets et les motifs séquentiels.

La section 2 introduit la problématique des représentations condensées ainsi que quelques définitions nécessaires pour la bonne compréhension de ce chapitre. La section 3 présente les travaux antérieurs pour les représentations basées sur les itemsets puis celles basées sur les motifs. Nous concluons ce chapitre par une discussion.

2 Problématique

Dans un monde qui s'ouvre de plus en plus facilement aux nouvelles technologies, il devient très facile de collecter et de stocker des informations. Or notre capacité d'extraction des *pépites de connaissances* reste quand à elle très limitée : soit les données sont trop denses et ne permettent pas des extractions avec des supports très bas pouvant faire apparaître des motifs ayant une forte valeur ajoutée, soit l'ensemble de motifs extraits est trop grand pour être d'une quelconque utilité pour l'analyste ou le décideur. C'est afin de répondre à ces problèmes que les chercheurs se sont penchés sur des représentations dites *condensées* afin de limiter le nombre de motifs extraits. Ces représentations condensées ont pour tâche de réduire la cardinalité de l'ensemble des motifs fréquents tout en gardant la même puissance d'expression que ce dernier. il existe ainsi tout un

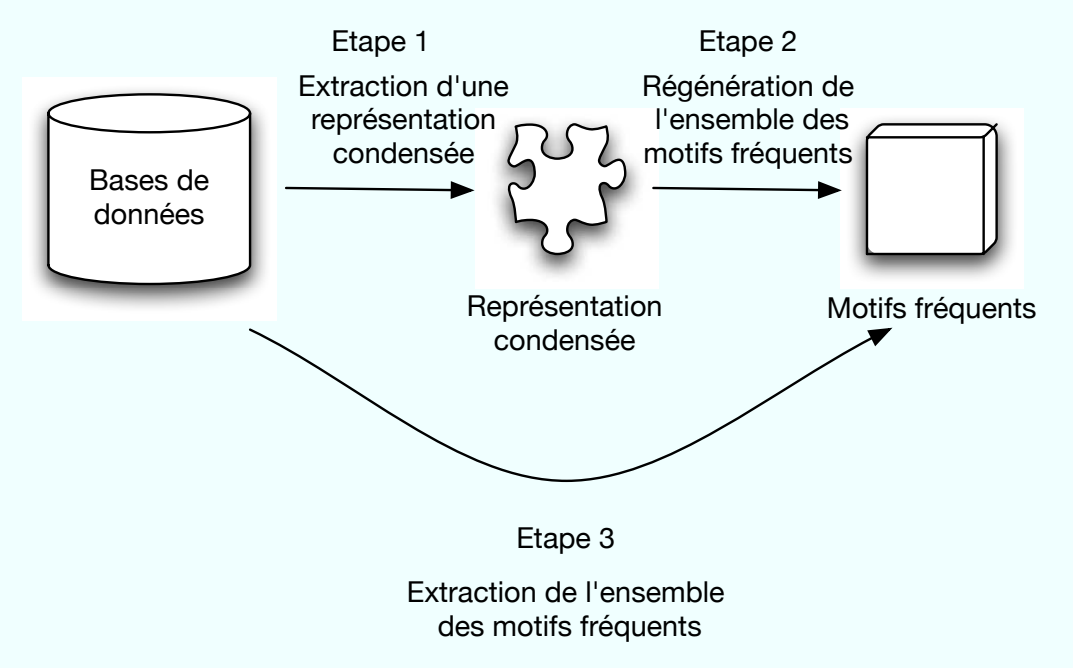


Fig. 3.1: Les différentes étapes du processus d'extraction de représentations condensées

processus permettant à partir d'une représentation condensée de régénérer l'ensemble de motifs fréquents (ou de le compléter).

La définition suivante donne un cadre plus formel pour les représentation condensée.

Définition 3.1 (Représentation condensée)**.** Soit $\mathcal{MF}(\mathcal{D}, \sigma)$ l'ensemble des motifs fréquents dans la base de données $\mathcal{D}$ pour un support minimal σ (dans le cadre de l'extraction des motifs séquentiels cet ensemble est noté $FSeqs(\mathcal{D}, \sigma)$ tout au long de ce manuscrit). Une représentation condensée $\mathcal{C}(\mathcal{D}, \sigma)$ est un ensemble contenant des motifs fréquents tel que :

$$\mathcal{C}(\mathcal{D}, \sigma) \subset \mathcal{MF}(\mathcal{D}, \sigma) \tag{3.1}$$

De plus, tous les éléments de $\mathcal{MF}(\mathcal{D}, \sigma) \setminus \mathcal{C}(\mathcal{D}, \sigma)$ doivent être régénérés de manière efficace à partir de $\mathcal{C}(\mathcal{D}, \sigma)$ uniquement et sans accès à la base de données $\mathcal{D}$.

Les premiers travaux sur les représentations condensées ont été introduits par H. Mannila et H. Toivonen dans [MT96] en se basant sur le concept de *calcul de théorie* [MT97]. Les auteurs présentent ainsi un ensemble de travaux formels permettant la construction de représentation condensées dans un cadre précis appelé l'ϵ-adéquation.

Il existe deux types de représentations condensées : les représentations dites *approximatives* et celles dites *exactes*. En fait si lors de l'opération de régénération de l'ensemble des motifs fréquents à partir d'une représentation condensée, nous ne pouvons qu'approximer (selon un certain taux

d'erreur) la valeur de support[1] alors nous parlons d'une représentation *approximative* (comme par exemple les représentations par motifs maximaux). Dans le cas contraire (une étape de régénération permettant une régénération exacte de la valeur de support) cette représentation condensée est dite *exacte*.

Un des aspects les plus importants pour la construction de représentations condensées est l'utilisation des bordures. Il existe deux types de bordures : (i) la bordure négative, notée $\mathcal{B}d^{-}(\mathcal{MF}(\mathcal{D},\sigma))$, est la collection de motifs qui contient les plus petits motifs (par rapport à la relation d'inclusion définie sur le motif) qui ne sont pas présents dans $\mathcal{MF}(\mathcal{D},\sigma)$ et (ii) la bordure positive, notée $\mathcal{B}d^{+}(\mathcal{MF}(\mathcal{D},\sigma))$, qui est la collection qui contient les plus grand motifs fréquents (i.e. l'ensemble des motifs fréquents maximaux).

Dans la prochaine section, nous présentons les différents travaux sur les représentations condensées en les analysant selon trois critères :

1. **La taille de la représentation** : plus une représentation est petite, plus elle est intéressante car on peut imaginer qu'il y aura moins de redondances.
2. **L'efficacité de l'approche** : en principe, une approche permettant d'extraire une représentation condensée doit être plus efficace en terme de temps de calcul et d'espace mémoire utilisé que l'approche d'extraction de tous les motifs fréquents. Nous illustrons ce critère dans la Figure 3.1 : les étapes 1 et 2 doivent être plus rapides et moins coûteuses en mémoire que l'étape 3.
3. **La complétude et efficacité de la régénération** : la régénération doit permettre de passer de la représentation condensée à l'ensemble de **tous** les motifs fréquents. Mais l'approche de régénération doit aussi être efficace en temps de calcul. En d'autres termes, l'étape 2 de la Figure 3.1 doit être plus rapide que l'étape 3.

3 Etat de l'art

Dans cette section, nous présentons les différentes approches condensées. Tout d'abord, nous nous focaliserons sur les représentations condensées dans le cadre de l'extraction d'itemsets puis nous aborderons les représentations dans le cadre d'extraction des motifs séquentiels.

[1]Selon le calcul de théorie [MT97], il peut y avoir plusieurs contraintes primitives, dans ce mémoire nous nous concentrons sur la contrainte de support/fréquence minimale.

3.1 Les représentations condensées dans le cadre de l'extraction d'itemsets

La recherche de représentations condensées a donnée lieu à de nombreux travaux [PBTL99, BB00, CG07, CG03, BBR03, EHZ05, SCR04]. Nous présentons ci-dessous les principales approches existantes ainsi que la théorie unificatrice des ensembles k-libres.

Représentation par itemsets maximaux

Si un itemset I est fréquent et qu'il n'existe pas d'itemsets fréquents J tels que $I \subset J$, alors l'itemset I est dit *maximal*. La représentation condensée par itemsets maximaux est une représentation approximative puisqu'elle ne permet pas de calculer précisément le support des itemsets mais de dériver une borne inférieure sur le support d'un itemset. Cette représentation se base sur la définition de bordure positive.

Nous illustrons cette représentation avec l'exemple suivant.

ID Transactions	Items
T_1	$\{a, b, c\}$
T_2	$\{b, c\}$
T_3	$\{b, c, e\}$
T_4	$\{b, c, d, e\}$
T_5	$\{a, b, c, d, e\}$

Tab. 3.1: Base de données $\mathcal{D}$ exemple contenant 5 transactions.

Exemple 3.1. Soit $\mathcal{D}$, la base de données transactionnelles représentée Table 3.1 et soit $\sigma = 3$, la valeur du support minimal pour l'extraction d'itemsets. L'ensemble des itemsets fréquents est :

$$\mathcal{F}_I(\mathcal{D}, \sigma) = \{(b) : 5, (c) : 5, (e) : 3, (bc) : 5, (be) : 3, (ce) : 3, (bce) : 3\}$$

Selon la définition proposée ci-dessus, la représentation contenant tous les itemsets fréquents maximaux est :

$$\mathcal{MF}_I(\mathcal{D}, \sigma) = \{(bce) : 3\}$$

A partir de $\mathcal{MF}_I(\mathcal{D}, \sigma)$, l'utilisateur ne peut pas régénérer de manière précise tous les supports des itemsets fréquents inclus dans (bce), mais grâce à la propriété d'anti-monotonie, l'utilisateur peut déduire, par exemple, que comme $(e) \subset (bce)$ alors $Support(e) \geqslant Support(bce)$ et approximer de cette manière le support des éléments dans l'ensemble $\mathcal{MF}_I(\mathcal{D}, \sigma)$.

Il existe plusieurs algorithmes d'extractions d'itemsets maximaux. A notre connaissance le premier algorithme fut celui presenté par Bayardo [Bay98] suivi d'autres algorithmes et implémentations efficaces tels que [BCF+05, GZ05].

Représentation par itemsets clos

La représentation basée sur les itemsets clos est une représentation exacte. Elle se base sur la notion des ensembles clos et tire son origine de la théorie des treillis et plus précisément des travaux autour de l'analyse de concepts formels [GW97]. Ainsi, un itemset I est dit *clos* si et seulement si il n'existe pas d'items tels que $I \subset J$ et $Support(I) = Support(J)$. En d'autres termes, si I n'a pas de super-itemset de même support. L'ensemble des itemsets fréquents clos est noté $\mathcal{CF}_I(\mathcal{D}, \sigma)$.

Propriété 3.1. [BTP+00] Un opérateur de fermeture divise l'ensemble des itemsets fréquents en classes d'équivalences disjointes par rapport à la propriété d'appartenance à la même fermeture. Ainsi, pour chaque classe d'équivalence, il existe :

1. Un *unique* plus grand élément, noté ifc et appelé l'itemset fréquent clos.
2. *Plusieurs* petits éléments appelés les générateurs minimaux fréquents de ifc.

En se basant sur cette propriété, il est possible de régénérer de manière exacte l'ensemble des itemsets fréquents à partir uniquement de l'ensemble des itemsets fréquents clos extraits au préalable : ainsi, si l'itemset I ne possède pas de super-itemset contenu dans la collection $\mathcal{CF}_I(\mathcal{D}, \sigma)$, alors I n'est pas fréquent. Si I possède un super-itemset $J \in \mathcal{CF}_I(\mathcal{D}, \sigma)$ alors : $Support(I) = Support(J)$. Cette notion de régénération exacte est illustrée dans l'exemple suivant.

Exemple 3.2. Soit $\mathcal{D}$, la base de données transactionnelle représentée Table 3.1 et soit $\sigma = 3$, la valeur du support minimal pour l'extraction d'itemsets. On remarque que l'itemset (be) n'est pas clos puisqu'il possède un super-itemset (bce) de même support 3.

L'ensemble des itemsets fréquents clos est :

$$\mathcal{CF}_I(\mathcal{D}, \sigma) = \{(bc) : 5, (bce) : 3\}$$

En possédant uniquement cette collection d'itemset l'analyste peut régénérer complètement la collection $\mathcal{F}_I(\mathcal{D}, \sigma)$. Par exemple générer le support de l'itemset (c), il suffit de trouver la fermeture la plus petite pour cet itemset : (bc), d'ou $Support(c) = 5$.

Les premiers travaux à avoir proposé une méthode pour calculer une représentation condensée close dans le cadre de l'extraction des itemsets sont [BB00] et [PBTL99].

Représentation par itemsets libres

Les itemsets libres (ou δ-libre) ont été introduits par J.F. Boulicaut et al. dans [BBR00, BBR03] et permettent d'avoir une représentation condensée approximative. Un itemset I est δ-libre, avec $\delta \in \mathbb{N}$, si pour chacun de ses sous-itemsets $J \subset I$, $Support(I) + \delta < Support(J)$.

L'ensemble des itemsets δ-libres fréquents est noté $\mathcal{F}\delta_I(\mathcal{D}, \sigma)$. Cette collection permet d'approximer les supports des itemsets qui ne sont pas δ-libres. Ainsi, si un itemset J n'est pas δ-libre, son support peut être approximé par le sous-itemset ayant le plut petit support dans l'ensemble des sous-itemsets δ-libres de J.

Exemple 3.1. Soit $\mathcal{D}$, la base de données transactionnelles représentée Table 3.1, soit $\sigma = 3$ et $\delta = 1$. L'ensemble des itemsets 1-libres fréquents est :

$$\mathcal{F}\delta_I(\mathcal{D}, \sigma) = \{(b) : 5, (c) : 5, (e) : 3\}$$

Les itemsets (bc), (be) et (ce) quand à eux ne sont pas pas 1-libre puisque : $Support(bc) + 1 > Support(c)$, $Support(be) + 1 > Support(e)$ et que $Support(ce) + 1 > Support(e)$. De plus, (bce) n'est pas 1-libre puisque si un itemset n'est pas δ-libre, alors par la propriété d'anti-monotonie des δ-libres, tous ses super-itemsets ne le sont pas aussi [BBR03].

Malheureusement, bien que les δ-libres permettent une représentation approximative de l'ensemble des itemsets non δ-libre (fréquents ou non fréquents), il est impossible en utilisant uniquement la collection $\mathcal{F}\delta_I(\mathcal{D}, \sigma)$ de décider si un itemset est fréquent ou pas. Pour résoudre cet inconvénient, les auteurs proposent de compléter la représentation en rajoutant une bordure qui dans ce cas est une collection représentant les itemsets minimaux δ-libres non fréquents.

Représentation par itemsets non-dérivables

La représentation par itemsets non-dérivables a été introduite par T. Calders et B. Goethals dans [CG02a]. Cette représentation se base sur des règles de déductions sur le support d'un itemset. Ainsi pour tous les itemsets $J \subseteq I$:

$$Support(I) \leqslant \sum_{J \subseteq X \subset I} (-1)^{|I \setminus X|+1} Support(X), \quad \text{si } |I \setminus J| \text{ est impair} \tag{3.2}$$

$$Support(I) \geqslant \sum_{J \subseteq X \subset I} (-1)^{|I \setminus X|+1} Support(X), \qquad \text{sinon.} \tag{3.3}$$

Ces règles sont construites en se basant sur un principe issu de l'analyse combinatoire : le principe de l'inclusion-exclusion. Ainsi, selon la parité (selon le sens arithmétique) de $|I \setminus J|$, l'itemset J produit soit une règle énonçant une borne inférieure sur le support, ou une borne supérieure. Étant donné un itemset I et l'ensemble de ses sous-itemsets, nous pouvons alors déduire

un intervalle sur son support noté $[LB(I), UB(I)]$ avec $LB(I)$ la borne inférieure et $UB(I)$ la borne supérieure. $|I \setminus J|$ est appelée la profondeur de la règle. Un itemset I est dit *dérivable* si et seulement si $LB(I) = UB(I)$ (i.e. nous pouvons déduire exactement le support de l'itemset I à partir du support de ses sous-itemsets). Malheureusement, la nombre de règles possibles afin de dériver le support d'un itemset croît de manière exponentielle. Pour palier ce problème, les auteurs proposent de n'essayer de dériver en pratique qu'à une profondeur donnée k. Les meilleures bornes inférieures et supérieures calculées pour un itemset I jusqu'à la profondeur k sont notées : $LB_k(I)$ et $UB_k(I)$.

A partir de ce concept de dérivabilité, les auteurs dans [CG02a, Cal03] proposent de ne garder comme représentation condensée que la collection des itemsets non-dérivables fréquents en arguant que la collection des itemsets fréquents peut être régénérée de manière itérative en utilisant les règles 3.2 à partir de l'ensemble des itemsets non-dérivables. Nous illustrons les notions de dérivabilité et de collection d'itemsets non-dérivables dans l'exemple suivant :

Exemple 3.2. La collection des itemsets non-dérivables fréquents, noté $\mathcal{NDF}_I(\mathcal{D}, \sigma)$, pour la base de données transactionnelles $\mathcal{D}$ représentée Table 3.1 et $\sigma = 3$ est :

$$\mathcal{NDF}_I(\mathcal{D}, \sigma) = \{(b) : 5, (c) : 5, (e) : 3\}$$

Dans cette collection, on ne garde pas l'itemset (bc) car cet itemset est dérivable. Ainsi, en séléctionnant le sous-itemset vide, noté $\emptyset$ et le sous-itemset (b), nous avons les règles suivantes :

1. $-Support(\emptyset) + Support(b) + Support(c) \leqslant Support(bc)$, qui implique que $5 \leqslant Support(bc)$.
2. $Support(b) \geqslant Support(bc)$.

Avec ces deux règles nous pouvons *déduire* que $5 \leqslant Support(bc) \leqslant 5$. Cet itemset, tout comme les itemsets (be), (ce) et (bce), est donc dérivable.

Il a été montré de manière empirique dans [CG02a, Cal03] que cette représentation condensée exacte peut parfois concurrencer en terme de taille de représentation la représentation close.

Une théorie unificatrice : les *k-libres*

T. Calders et B. Goethals ont proposé dans l'article [CG03], une théorie regroupant sous un unique cadre formel plusieurs représentations condensées et leurs propriétés communes. Ces représentations condensées sont les itemsets non-dérivables, les itemsets 0-libres et les itemsets disjonctifs-libres. Cette dernière représentation a été introduite dans [BR01, BR03] et se base principalement sur le concept de règles disjonctives (i.e. $X \Rightarrow a \vee b$, avec $X \subseteq \mathcal{I}$ et $a, b \in \mathcal{I} \setminus X$).

Ainsi, un itemset $I = X \bigcup \{a, b\}$ est dit disjonctif-libre si et seulement si il n'existe pas de règle disjonctive telle que : $X \Rightarrow a \vee b$. Les itemsets δ-libres et les itemsets disjonctifs-libres sont un cas particulier de la représentation condensée dite des itemsets disjonctifs libres généralisés [KG02a, KG02b].

Cet ensemble de représentations condensées peut être exprimé en se basant sur un concept d'itemsets k-libres[1] fréquents couplés avec une bordure.

Définition 3.2 (Itemsets k-libres [CRB06])**.**

- Un itemset I est dit k-libre si $Support(I) \neq LB_k(I)$ et $Support(I) \neq UB_k(I)$.
- Un itemset I est dit ∞-libre si $Support(I) \neq LB(I)$ et $Support(I) \neq UB(I)$.

La propriété suivante issue de [CG03] exhibe le lien entre les trois représentations condensées : non-dérivables, 0-libres et disjonctifs-libres et les itemsets k-libres.

Propriété 3.2.

- Un itemset I est 0-libre (dans le sens δ-libre avec $\delta = 0$) si et seulement si I est 1-libre (dans le sens k-libre).
- Un itemset I est disjonctif-libre si et seulement si I est 2-libre (dans le sens k-libre).

La propriété d'antimonotonie ainsi que les discussions sur le type et la taille des bordures associées sont détaillées dans [CG03]. Ainsi l'unification de ces représentations condensées par itemsets k-libres permet en fait d'exhiber une relation hiérarchique : plus k est grand plus la représentation est concise. Bien sûr cette augmentation de l'efficacité (une représentation de plus en plus concise) s'accompagne d'une plus grande complexité. Ainsi, pour s'en convaincre, il suffit de voir que la représentation des 0-libres (dans le sens δ-libre avec $\delta = 0$) se base sur les itemsets 1-libres (dans le sens k-libre) alors que la représentation par itemset non-dérivables qui est plus concise se base quand à elle sur les itemsets ∞-libres (dans le sens k-libre).

3.2 Les représentations condensées dans le cadre de l'extraction de motifs séquentiels

Tout comme pour l'extraction des itemsets fréquents, l'extraction de séquences devient problématique si la base de données est trop dense ou si le seuil de support minimal est trop bas. Ainsi, en parallèle avec le développement de nouvelles représentations condensées des itemsets, des chercheurs se sont attelés à découvrir de nouvelles représentations condensées dans le cadre des motifs séquentiels. Malheureusement, seules deux représentations condensées ont été actuellement développées : la représentation approximative basée sur les motifs séquentiels maximaux et celle exacte basée sur les motifs séquentiels clos.

[1] A ne pas confondre avec les δ-libres.

3.3 Recherche des motifs séquentiels maximaux

Cette représentation à été introduite dans les premiers travaux de R. Srikant [Sri95], ainsi l'auteur présentait trois algorithmes dont deux permettaient l'extraction de motifs séquentiels maximaux. La définition d'un motif séquentiel maximal est similaire à celle des itemsets fréquents maximaux. Ainsi, si une séquence s est fréquente et qu'il n'existe pas de séquences fréquentes s' telles que $s' \prec s$, alors le motif séquentiel s est dit *maximal*.

Exemple 3.3. En utilisant la base de données transactionnelles présentée dans la Table 1.2 et un support minimal $\sigma = 3$, nous avons l'ensemble de motifs séquentiels maximaux suivant :

$$\mathcal{M}FSeqs(\mathcal{D}, \sigma) = \{\langle(d)\rangle : 3, \langle(b,c)\rangle : 4, \langle(c,e)\rangle : 4, \langle(a)(b,e)\rangle : 3, \langle(c)(b,e) : 3\rangle\}$$

Comme dans le cadre de l'extraction de motifs séquentiels, cette représentation ne permet qu'une approximation, souvent grossière, du support des séquences fréquentes mais elle reste particulièrement intéressante dans des jeux de données très denses comme les jeux de données biologiques.

3.4 Recherche des motifs séquentiels clos

Les auteurs de [YHA03] soulignent la necessité d'une représentation condensée basée sur les motifs séquentiels clos en utilisant l'exemple d'une base de données transactionnelles ne contenant qu'un seul motif : $\langle(a_1)\ (a_2)\ \ldots(a_{100})\rangle$. Dans ce cas, il faudra générer $2^{100} - 1$ sous-séquences fréquentes ayant toutes un support minimum de 1. Ces sous-séquences seront redondantes car elles auront toutes le même support que $\langle(a_1)\ (a_2)\ \ldots\ (a_{100})\rangle$. Ainsi, dans [YHA03], les auteurs définissent donc la problématique de la recherche des motifs séquentiels clos (*closed sequential patterns*) en s'inspirant de la recherche d'itemsets clos. Or, contrairement au cadre des itemsets, les auteurs dans les différents travaux sur les séquences closes n'exhibent pas de liens directs avec la théorie des treillis et l'analyse de concepts formels en présentant des opérateurs de fermetures, mais se basent uniquement sur une définition simplifiée.

Définition 3.3. Soit σ, le support minimum et $FSeqs(\mathcal{D}, \sigma)$ l'ensemble des motifs séquentiels fréquents correspondants. L'ensemble des motifs séquentiels clos $\mathcal{C}FSeqs(\mathcal{D}, \sigma)$ est défini comme :

$$\mathcal{C}FSeqs(\mathcal{D}, \sigma) = \{s \mid s \in FSeqs(\mathcal{D}, \sigma) \wedge \nexists\ s' | s \preceq s' \text{ avec } Support(s) = Support(s')\}$$

Exemple 3.4. En utilisant la base de données transactionnelles présentée dans Table 1.2 et un support minimal $\sigma = 3$, l'ensemble des motifs séquentiels clos est :

$$\mathcal{C}FSeqs(\mathcal{D}, \sigma) = \{\langle(d)\rangle : 3, \langle(b,c)\rangle : 4, \langle(b,e)\rangle : 5, \langle(c,e)\rangle : 4, \langle(a)(b,e)\rangle : 3, \langle(c)(b,e)\rangle : 3\}$$

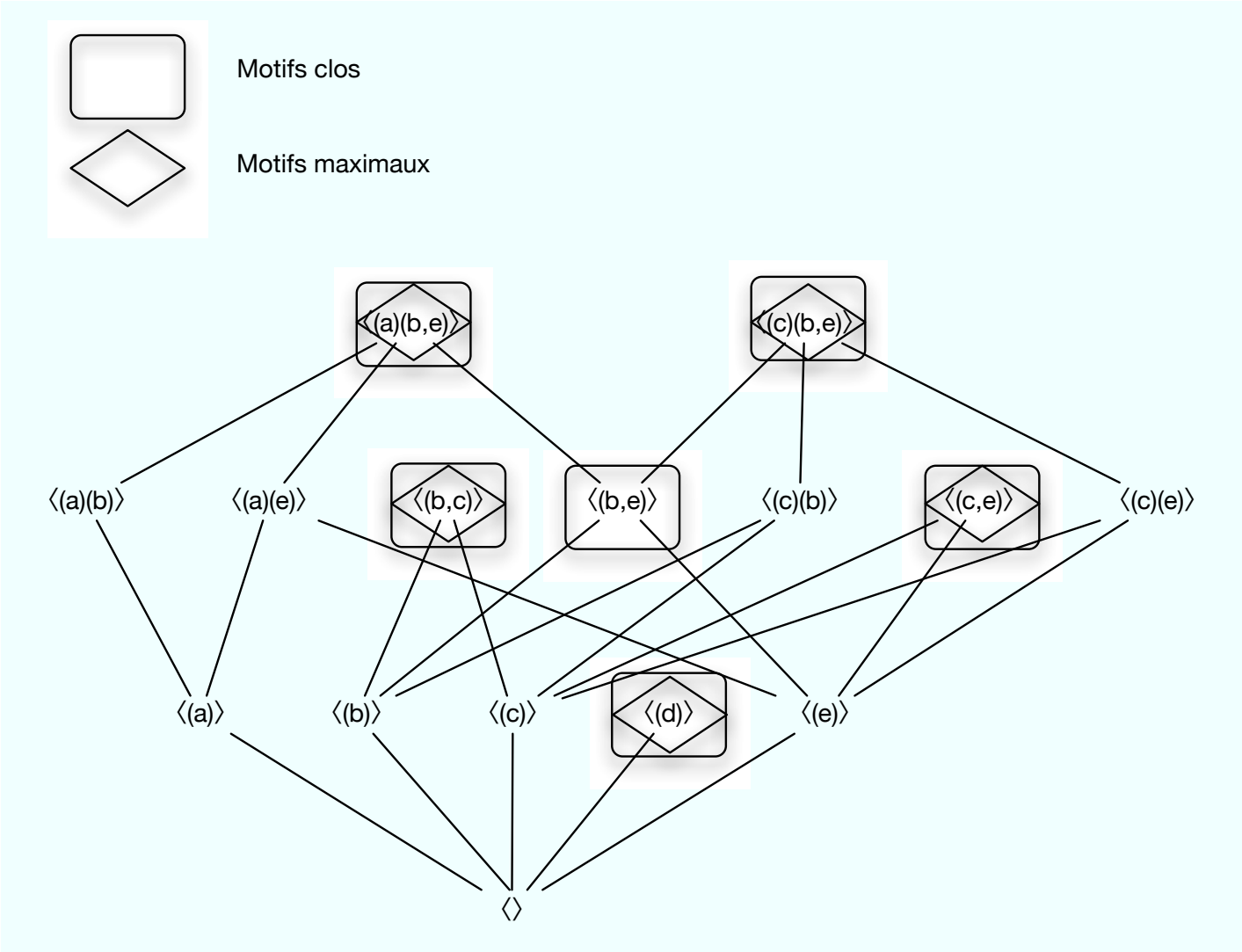


Fig. 3.2: Motifs fréquents clos et maximaux

La Figure 3.2 montre les différences entre les deux représentations closes et maximales dans le cadre de cet exemple.

Il est remarquable, dans le cadre de cette représentation pour les séquences, de noter que peu de travaux se sont penchés sur les bases théoriques des concepts de *fermetures*. Ceci est dû premièrement à la grande difficulté de développer une branche de la théorie des treillis dans le cadre des séquences (ou de n'importe quel motif structuré). Ainsi, les auteurs des différents articles sur la représentation close pour les motifs séquentiels se concentrent uniquement sur les heuristiques ou les types de structures de données permettant d'optimiser l'opération de fouille et passent généralement sous silence le coté théorique. Notons ainsi qu'il existe deux algorithmes d'extraction de motifs séquentiels, (*i*) CloSpan, qui fut le premier algorithme capable de résoudre ce problème et (*ii*) dans [WH04], l'approche BIDE qui utilise une nouvelle manière d'étendre les séquences et optimise l'espace de recherche en analysant à l'avance les motifs pouvant être étendus.

CLOSPAN

CloSpan [YHA03] est une méthode basée sur le principe depth-first et implémente l'algorithme PrefixSpan. En fait, il s'agit d'une optimisation de ce dernier, destinée à élaguer l'espace de recherche en évitant de parcourir certaines branches dans le processus de divisions récursives (en détectant par avance les motifs séquentiels non clos). Le principe de CloSpan repose sur deux éléments essentiels : l'ordre lexicographique des séquences et la détection de liens systématiques entre deux items (i.e."β apparaît toujours avant γ dans la base de données").

BIDE

CloSpan ne s'avère pas efficace dans le cas de bases contenant de trop nombreuses séquences fermées. Pour pallier ce problème, une nouvelle approche, BIDE (BI-Directional Extension) est proposée dans [WH04]. L'idée générale est d'étendre les séquences dans les deux directions, i.e. en avant (*forward extension*) et en arrière (*backward extension*). En effet, considérons une séquence S = $i_1 i_2 ... i_n$, celle-ci peut être étendue de trois manières possibles :

1. ajout d'un item après i_n
2. ajout d'un item entre $i_1, i_2 \dots i_n$
3. ajout d'un item avant i_1

La première correspond à une extension en avant et les deux dernières à une extension en arrière. Ainsi, les auteurs montrent que pour une séquence s, s'il n'y a pas d'extension avant ni d'extension arrière alors s est une séquence close. Comme dans CloSpan, une base projetée est constituée. Pour une séquence s, son ensemble d'items extensibles en avant, i.e. les items qui peuvent être ajoutés

	Taille	Efficacité	Complétude
Itemsets maximaux	•••	••	approximative
Itemsets clos	••	•••	exacte
Itemsets libres	••	••	approximative et nécessite une bordure
Itemsets non-dérivables	••	•••	nécessite des calculs de dérivations de supports

Tab. 3.2: Synthèse des approches pour les représentations condensées des itemsets

	Taille	Efficacité	Complétude
Séquences maximales	•••	••	approximative
Séquences closes	••	•••	exacte

Tab. 3.3: Synthèse des approches pour les représentations condensées des séquences

à la fin de s, est constitué par les items locaux dont le support est égal à celui de la séquence. Ces items locaux sont simplement trouvés en parcourant la base projetée pour ce préfixe et en comptant le nombre d'items. Pour effectuer rapidement cette opération, la projection utilisée est une pseudo projection comme dans [PHW02]. De manière à définir les extensions possibles en arrière, il faut dans un premier temps rechercher, pour les items d'une séquence, quelles sont les extensions en arrière possibles. Pour cela, il est nécessaire de remonter dans la séquence pour examiner avec quel item il est possible de l'étendre [WH04].

4 Discussion

Dans ce chapitre, nous avons survolé les différentes représentations condensées possibles dans le cadre de l'extraction des itemsets et des motifs séquentiels. Les tableaux 3.2 et 3.3 résument les différentes approches présentées précédemment en fonction des critères d'analyse définis (taille, efficacité, complétude de la régénération).

Malheureusement, alors qu'il existe plusieurs représentations condensées dans le cadre de l'extraction des itemsets, il n'en existe que deux qui sont définies dans le cadre des motifs séquentiels. *Pourquoi est-ce si difficile ?* Nous verrons dans le chapitre suivant que la raison essentielle est liée au fait que les fondements mathématiques utilisés dans le cadre des représentations condensées pour les itemsets ne sont pas facilement transposables aux motifs structurés comme les séquences. Ainsi, dans le cadre, par exemple, de la représentation par séquences fréquentes closes, les chercheurs se sont plus penchés vers le coté pratique que vers le coté théorique. Cela s'explique par la

difficulté de définir des connections de Galois pour les relations binaires basées sur des séquences. Le seul travail ayant abouti dans ce sens est à notre connaissance le travail de G. C. Garriga qui a réussi à définir un cadre formel théorique précis permettant de définir des opérateurs de fermetures pour les motifs séquentiels [CG06].

Mais qu'en est-il des autres représentations condensées ? Est-il possible, comme pour les itemsets de dériver des règles sur les supports de séquences et ainsi éliminer certaines redondances ? Nous donnerons des réponses à ces questions dans le chapitre suivant.

Chapitre 4

Règles de dérivations pour la fréquence des motifs séquentiels

Le problème d'extraction des motifs séquentiels souffre encore plus que l'extraction des itemsets fréquents du nombre impressionnant de motifs que le processus de fouille peut extraire. Ainsi, il n'est pas rare de voir que l'extraction d'une petite base de données produit des dizaines de millions de motifs fréquents. Or généralement, le but de la fouille de données est d'extraire des *pépites de connaissances* noyées dans un magma confus d'informations inutiles. De fait, cette situation est une véritable *contradictio in terminis* : d'un coté, le but de la fouille est de chercher les connaissances rares mais intéressantes et de l'autre nous nous retrouvons avec un ensemble de motifs impossible à analyser humainement. L'objectif de ce chapitre est de proposer justement une solution à ce problème dans le cadre des motifs séquentiels.

Une partie des travaux présentés dans ce chapitre a été publiée dans la conférence *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD 2008)* et le journal *Data Mining and Knowledge Discovery Journal*.

1 Introduction

Afin de résoudre le paradoxe des résultats inintéressants *noyant les pépites de connaissances*, un ensemble de chercheurs se sont attelés, dans le domaine de l'extraction des itemsets, à éliminer la redondance dans les résultats obtenus lors du processus de fouille en proposant des représentations condensées des itemsets fréquents [CRB06]. Une approche logique, serait de voir comment ces représentations pourraient être étendues des itemsets fréquents au domaine des motifs.

En effet, et comme nous l'avons déjà souligné dans le chapitre 3 de la partieI, seules les représentations maximale et close existent pour les motifs séquentiels [YHA03, WH04]. Dans ce

chapitre, nous analysons la possibilité d'étendre la représentation des itemsets non-dérivables pour le domaine des séquences. Un itemset dérivable est un itemset dont le support est parfaitement déterminé par ses sous-itemsets. Un itemset dérivable peut donc être considéré comme redondant et être éliminé du résultat final. Pourtant, l'élégance et la simplicité de cette définition dans le cadre des itemsets cache un problème autrement plus difficile pour les motifs séquentiels. Ainsi, nous allons montrer dans ce chapitre qu'il ne peut pas exister de séquences non-dérivables.

Contrairement donc aux itemsets, la notion de dérivabilité pour les séquences est inintéressante voir même inutile. Ce résultat négatif nous permet d'énoncer que **certaines approches de représentations condensées basées sur le support (ou la fréquence) issues du domaine des itemsets sont impossibles à appliquer dans le cadre des séquences**.

Afin d'énoncer ce résultat, nous étudions dans ce chapitre le problème central `SEQFREQSAT`. Ce problème est défini comme le choix de décider si un ensemble de contraintes sur des fréquences de motifs séquentiels est satisfiable (en exhibant par exemple une base de données respectant les contraintes de fréquences). Ce problème est une extension du problème théorique introduit par T. Calders dans [Cal03]. Nous montrons ainsi que ce problème demeure **NP**-complet dans le cadre des motifs séquentiels. Ce résultat ne permet somme toute pas une utilisation directe dans le domaine de la fouille de motifs séquentiels mais nous présentons un cas spécial de complexité moindre et intéressant d'un point de vue pratique qui permet d'ouvrir de nouvelles voies vers des algorithmes d'extractions plus rapides.

Notre contribution dans ce chapitre est triple :

1. Nous présentons une nouvelle relation d'équivalence et de classe d'équivalence pour les motifs séquentiels.
2. Nous introduisons la problème `SEQFREQSAT` et discutons sa complexité et son utilisation dans le cadre de la fouille de motifs séquentiels. Nous discutons le rôle des classes d'équivalences de motifs séquentiels et du problème `SEQFREQSAT` dans les représentations condensées pour les motifs séquentiels.
3. Nous présentons un cas spécial (et pratique) du problème `SEQFREQSAT` qui permet de calculer une borne supérieure sur la fréquence d'une séquence uniquement à partir de ces sous-séquences et sans passes sur la base de données.

Le chapitre est organisé de la manière suivante. La section 2 énonce la problématique de ce chapitre et le lien avec les représentations condensées dans le domaine des itemsets. La section 3 présente les différents concepts préliminaires nécessaires à la bonne compréhension de ce chapitre. Dans la section 4 nous introduisons les classes d'équivalences associées aux motifs séquentiels et leurs propriétés. Le problème `SEQFREQSAT` est introduit et discuté dans la section 5. La section 6 discute le cas spécial du problème `SEQFREQSAT` et les différentes méthodes de calcul de bornes supérieures sur la fréquence d'une séquence. Nous concluons ce chapitre par une discussion.

2 Énoncé de la problématique

Le problème étudié dans ce chapitre peut être énoncé de la façon suivante : généralement pour plusieurs bases de données transactionnelles $\mathcal{D}$ avec des seuils σ de fréquences ou de supports différents, la taille de l'ensemble des séquences fréquentes $FSeqs(\mathcal{D}, \sigma)$ est extrêmement large et contient beaucoup de redondances. Le but de ce chapitre est d'étudier la possibilité de réduire l'ensemble des motifs fréquents en se basant sur des approches condensées utilisées dans le domaine des itemsets. L'approche par itemsets clos a déjà été appliquée au domaine des séquences [YHA03]. Ainsi l'extraction de motifs séquentiels clos permet d'avoir généralement un ensemble de résultats beaucoup plus petit que $FSeqs(\mathcal{D}, \sigma)$. Dans ce chapitre nous voulons étendre une autre classe de méthodes de représentations condensées d'itemsets pour les séquences : les ensembles k-libre [CG03]. Cette classe inclus les ensembles libres, les ensembles disjoints libres et les itemsets non-dérivables. Cette classe de méthodes de représentations condensées d'itemsets se base sur le concept de déduction de fréquences. Ainsi, pour un ensemble donné de fréquences, nous nous posons la question de savoir quelles sont les autres fréquences que l'on peut déduire. Ce problème est formalisé de la manière suivante :

Soit $\mathcal{C} = \{f_{S_1}, \ldots, f_{S_n} \in [0,1]\}$ un ensemble de fréquences pour $S_1, \ldots, S_n$. Une base de données transactionnelles est dite *consistante* avec $\mathcal{C}$ si et seulement si, pour $i = 1 \ldots n$, $f_{S_i,\mathcal{D}} = f_{S_i}$. Généralement, pour un ensemble de fréquences donné, il existe plusieurs bases de données consistantes. Supposons maintenant S une séquence. Les *meilleures bornes* pour f_S, étant données les n fréquences sont définies de la manière suivante :

$$[LB_{\mathcal{C}}(S), UB_{\mathcal{C}}(S)] := [\min\{f_{S,\mathcal{D}} \mid \mathcal{D} \text{ consistant avec } f_{S_1}, \ldots, f_{S_n}\}, \max\{f_{S,\mathcal{D}} \mid \mathcal{D} \text{ consistant avec } f_{S_1}, \ldots, f_{S_n}\}]$$

Dans le domaine de l'extraction des itemsets fréquents, des *règles de déductions* basées sur les bornes et quelques conditions sur l'ensemble $\mathcal{C}$ ont été étudiées. Par exemple, pour l'itemset (abc), nous avons les règles suivantes : $f_{abc} \leqslant f_{ab}$ et $f_{abc} \geqslant f_{ab} + f_{ac} - f_a$. Supposons maintenant que la borne inférieure f_{ab} soit égale à la borne supérieure $f_{ab} + f_{ac} - f_a$, alors (abc) est dans ce cas redondant par rapport aux itemsets (a), (ab), et (ac). Ainsi, dans ce cas bien précis, il devient possible d'éliminer (abc) de l'ensemble de résultats final. Malheureusement, le problème du choix des meilleures bornes est **NP**-difficile [Cal08], mais pour certains cas spéciaux, ce problème peut être utilisé dans des cas pratiques. Un de ces cas spéciaux forme la base de la représentation en itemsets non-dérivables. La question fondamentale que nous nous posons dans ce chapitre est : *Peut-on avoir des règles de déductions pour les motifs séquentiels ?* Dans le cas d'une réponse positive, *peut-on avoir une représentation condensée basée sur des séquences non-dérivables ?*

3 Concepts préliminaires et définitions

Dans cette section nous rappelons quelques définitions sur l'extraction de motifs séquentiels ainsi que des définitions issues de la théorie des treillis.

3.1 Extraction de motifs séquentiels

Nous rappelons ici la définition 2.5 avec quelques légères modifications nécessaires à la bonne compréhension des formalismes introduits dans ce chapitre.

$\mathcal{T}(\mathcal{I})$ représente l'ensemble (infini) de toutes les séquences possibles à partir des items présents dans l'ensemble $\mathcal{I}$. Une base de données transactionnelles $\mathcal{D}$ basée sur les items de $\mathcal{I}$ est un ensemble fini de paires (SID, T), appelées transactions, avec $SID \in \{1, 2, \ldots\}$ un identifiant et $T \in \mathcal{T}(\mathcal{I})$ une séquence construite sur $\mathcal{I}$. Pour 2 transactions quelconques $(SID_1, T_1) \neq (SID_2, T_2) \in \mathcal{D}$ implique que $SID_1 \neq SID_2$.

Définition 4.1 (Support)**.** Le support d'une séquence S dans une base de données transactionnelles $\mathcal{D}$, noté $Support(S, \mathcal{D})$, est défini tel que :

$$Support(S, \mathcal{D}) = |\{(SID, T) \in \mathcal{D} | S \preceq T\}| \tag{4.1}$$

La fréquence de S dans $\mathcal{D}$, noté f_S, est définie telle que :

$$f_S = \frac{Support(S, \mathcal{D})}{|\mathcal{D}|} \tag{4.2}$$

3.2 Quelques éléments de la théorie des ensembles et des treillis

En mathématiques, comme dans la vie courante, il y a des ensembles contenant certains éléments (couples, triplets, etc...) reliés les uns aux autres par une propriété particulière que d'autres éléments de l'ensemble ne présentent pas. Une partie des démonstrations et des preuves décrites dans ce chapitre utilise quelques éléments de la théorie des treillis introduite dans [GW97].

Définition 4.2 (Relation Binaire)**.** Une relation binaire R entre deux ensembles M et N, notée mRn, est un ensemble de paires (m, n) tel que $m \in M$ et $n \in N$. Si $N = M$, nous parlons alors d'une relation binaire sur l'ensemble M.

Définition 4.3 (Ordre)**.** Une relation binaire R sur un ensemble M est appelée relation d'ordre, si pour tout $x, y, z \in M$ elle satisfait les conditions suivantes :

1. xRx (réflexivité)

2. xRy et $yRx \Rightarrow x = y$ (antisymétrie)
3. xRy et $yRz \Rightarrow xRz$ (transitivité)

Si une relation d'ordre, notée $\leqslant$, est définie sur un ensemble M, on dit alors que M est un *ensemble ordonné* relativement à $\leqslant$. Si, de plus pour tout couple (x, y) de M, l'une au moins des relations $x \leqslant y$ ou $y \leqslant x$ a lieu, on dit que M est un *ensemble totalement ordonné*, ou une chaîne.

Exemple 4.1. Un exemple simple et très connu est l'ensemble $\mathbb{R}$ des nombres réels ; celui-ci est totalement ordonné par la relation $\leqslant$ définie par les couples (x, y) tels que $y - x$ soit positif. Cette relation est appelée *ordre naturel* des nombres réels.

Définition 4.4 (Équivalence)**.** Une relation R définie sur l'ensemble M est appelée *relation d'équivalence* si elle est réflexive, symétrique et transitive. Il est évident que la relation R définie par $xRy \Leftrightarrow x = y$ appelée *relation d'égalité*, est la seule relation qui soit à la fois une relation d'ordre et une relation d'équivalence.

Définition 4.5 (Bornes supérieures et inférieures)**.** Soit L une partie non-vide d'un ensemble ordonné M. On dit qu'un élément a est un *majorant* (respectivement un *minorant*) de L si, pour tout élément $x \in L$, $x \leqslant a$ (respectivement $x \geqslant a$). Si L a au moins un majorant (respectivement un minorant), on dit alors que L est une *partie majorée* (respectivement *partie minorée*) de M. Une partie majorée et minorée est dite *bornée*. Si $L = M$, alors nous parlons respectivement d'un *ensemble majoré*, d'un *ensemble minoré* ou d'un *ensemble borné*.

Soit de nouveau L une partie de l'ensemble ordonné M. L'ensemble des majorants et l'ensemble des minorants de M sont eux-mêmes des parties de M. Il s'ensuit que les définitions précédentes peuvent s'appliquer à ces deux parties. En conséquences nous introduirons les termes suivants : nous appellerons *borne inférieure* (respectivement *borne supérieure*) de L dans P le plus grand (respectivement le plus petit) élément, s'il existe de l'ensemble des minorants (respectivement des majorants) de M.

Il découle des considérations précédentes que toute partie de L de M admet au plus une borne inférieure et une borne supérieure. Celles-ci sont notées $\bigwedge L$ et $\bigvee L$.

Définition 4.6 (Couverture)**.** Soit (a, b) un couple d'éléments d'un ensemble ordonné tel que $a < b$ et qu'il n'existe aucun élément c tel que $a < c < b$. On dit alors que les éléments a et b sont consécutifs, que *a est couvert par b*. Cette situation est exprimée par le symbole $a \prec b$.

Définition 4.7 (Treillis)**.** Nous procédons ici à une définition se basant sur la relation d'ordre. Un treillis est un ensemble M muni d'une relation d'ordre $\leqslant$ vérifiant que pour tous les éléments

a et b de M, il existe une borne supérieure, notée $x \vee y$ (appelé union), et une borne inférieure, notée $x \wedge y$ (appelé intersection), à l'ensemble $\{a, b\}$.

Un treillis M est dit *complet* si et seulement si pour tout sous-ensemble L de M, L possède une borne supérieure et une borne inférieure

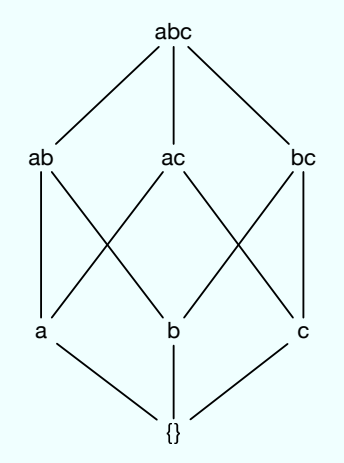


Fig. 4.1: Représentation graphique d'un treillis pour les parties de l'ensemble $\{a, b, c\}$

Exemple 4.2.
- L'ensemble des parties d'un ensemble muni de l'inclusion forme un treillis où la borne supérieure est l'union et la borne inférieure l'intersection (voir Figure 4.1).
- Dans l'ensemble $\mathbb{N}^*$ des entiers naturels non-nuls, notons $a \wedge b$ et $a \vee b$ le plus grand commun diviseur et le plus petit commun multiple, respectivement, des nombres a et b. Muni de ces 2 opérations $\mathbb{N}^*$ est un treillis.

Définition 4.8 (Hyper-treillis)**.** Un hyper-treillis est un ensemble M muni d'une relation d'ordre $\leqslant$ vérifiant que pour tous les éléments a et b de M, il existe un ensemble de bornes supérieures et un ensemble de bornes inférieures à l'ensemble $\{a, b\}$.

Observation 4.1. La relation d'inclusion $\preceq$ définit un hyper-treillis sur $\mathcal{T}(\mathcal{I})$, l'ensemble infini des séquences possibles sur $\mathcal{I}$.

Exemple 4.3. Supposons $\mathcal{L} = \{\langle\rangle, \langle a\rangle, \langle b\rangle, \langle (b)(a)\rangle, \langle (ab)\rangle, \langle (ab)(a)\rangle\}$ un sous-ensemble de $\mathcal{T}(\mathcal{I})$ avec $\mathcal{I} = \{a, b\}$. L'ensemble $\mathcal{L}$ muni de la relation d'inclusion $\preceq$ définit bien un hyper-treillis (voir Figure 4.2). On remarque bien que l'on n'a pas une unique borne supérieure mais un ensemble : $a \vee b = \{\langle (b)(a)\rangle, \langle (ab)\rangle\}$, de même pour la borne inférieure : $\langle (ab)\rangle \wedge \langle (b)(a)\rangle = \{\langle (b)(a)\rangle, \langle (ab)\rangle\}$.

L'ensemble des motifs séquentiels peut donc être représenté de manière algébrique sous la forme d'un hyper-treillis. Malheureusement, cette particularité rend l'utilisation directe des approches de

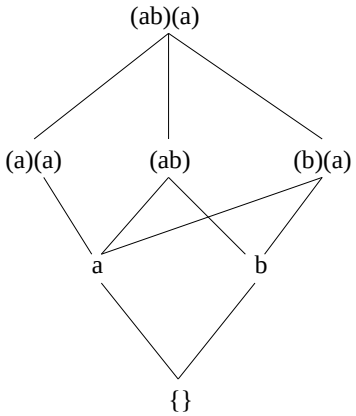


FIG. 4.2: Représentation graphique de $(\mathcal{L}, \preceq)$.

représentations condensées du domaine de l'extraction des itemsets difficiles, voire impossible car les unions et les intersections des séquences ne sont pas uniques (voir Exemple 4.3). Ainsi, pour les représentations condensées basées sur le support ou la fréquence nous devons prendre en compte **toutes** les séquences possibles qui supportent une certaine sous-séquence. Une question reste alors en suspens : *comment pouvons-nous caractériser ou regrouper les séquences qui supportent un ensemble de sous-séquences précis ?*

Nous répondons à cette question dans la section suivante.

4 Classes d'équivalences pour les motifs séquentiels

L'idée de l'utilisation de classes d'équivalences pour les motifs séquentiels fut introduite par M.J. Zaki dans [Zak01]. Ces classes, basées sur une relation de préfixe permettent de décomposer le problème d'extraction de motifs séquentiels en sous-problèmes plus simples. Cette approche utilisée dans l'algorithme SPADE permet entre autre de diviser l'espace de recherche et d'optimiser ainsi le processus d'extraction. Dans cette section, nous introduisons un concept de classes d'équivalences basé sur une relation d'équivalence définie sur la notion de support. Nous discutons de l'utilité de ces classes d'équivalences dans le domaine de l'extraction de séquences et nous étudions les problèmes de complexités associés.

4.1 Définitions

Définition 4.9. Soit $\mathcal{S}$ un ensemble de séquences. Deux séquences T_1 et T_2 sont dites $\mathcal{S}$*-équivalentes*, noté $T_1 \equiv_{\mathcal{S}} T_2$, si, pour tout $S \in \mathcal{S}$ nous avons $S \preceq T_1$ si et seulement si $S \preceq T_2$. L'ensemble de toutes les séquences équivalentes à $\mathcal{T}$ sous $\equiv_{\mathcal{S}}$ est noté $[T]_{\mathcal{S}}$.

Soit $\mathcal{IN}$ et $\mathcal{OUT}$, 2 ensembles de séquences. Alors $\mathcal{E}(\mathcal{IN}, \mathcal{OUT})$ définit l'ensemble de séquences tel que :

$$\mathcal{E}(\mathcal{IN}, \mathcal{OUT}) \ := \ \{T \in \mathcal{T} \mid \forall S \in \mathcal{OUT} : S \preceq T \Leftrightarrow S \in \mathcal{IN}\} \tag{4.3}$$

$\mathcal{E}(\mathcal{IN}, \mathcal{OUT})$ définit ainsi l'ensemble des séquences qui supportent toutes les séquences de l'ensemble $\mathcal{IN}$ mais ne supportent aucune séquence présente uniquement dans l'ensemble $\mathcal{OUT}$.

Exemple 4.4. Posons $\mathcal{I} = \{a, b, c\}$, et

$$\mathcal{S} = \left\{ \begin{array}{l} \langle (a)(b)(c) \rangle \\ \langle (b)(c) \rangle \\ \langle (a)(b) \rangle \end{array} \right\}$$

Nous avons alors : $\langle (abc)(abc)(abc) \rangle \equiv_{\mathcal{S}} \langle (a)(bc)(abc) \rangle$ et $\langle (b)(c) \rangle \not\equiv_{\mathcal{S}} \langle (a)(bc) \rangle$.

Considérons maintenant $\mathcal{E}(\mathcal{IN}, \mathcal{OUT})$:

- Soit $\mathcal{IN} = \{\langle (a)(b)(c) \rangle, \langle (ac) \rangle\}$ et $\mathcal{OUT} = \{\langle (bc) \rangle\}$.
  $\mathcal{E}(\mathcal{IN}, \mathcal{OUT})$ contient (mais pas uniquement) les séquences $\langle (ac)(b)(c) \rangle$, $\langle (ac)(b)(c) \rangle$. La séquence $\langle (abc)(b)(c) \rangle$ quant à elle n'est pas incluse dans $\mathcal{E}(\mathcal{IN}, \mathcal{OUT})$ puisque $\langle (ac) \rangle \preceq \langle (abc)(b)(c) \rangle$.

- Soit $\mathcal{IN} = \{\langle (ab) \rangle, \langle (ac) \rangle\}$ et $\mathcal{OUT} = \{\langle (c) \rangle\}$.
  $\mathcal{E}(\mathcal{IN}, \mathcal{OUT})$ est vide puisque pour chaque séquence contenant $\langle (ac) \rangle$ doit aussi contenir $\langle c \rangle$.

- $\mathcal{IN} = \{\langle (a)(c) \rangle, \langle (b) \rangle\}$ et $\mathcal{OUT} = \{\langle (a)(b) \rangle, \langle (b)(c) \rangle\}$.
  $\mathcal{E}(\mathcal{IN}, \mathcal{OUT})$ est aussi vide, car chaque séquence T qui contient $\langle (a)(c) \rangle$ et $\langle (b) \rangle$doit soit contenir $\langle (a)(b) \rangle$ ou $\langle (b)(c) \rangle$; comme T contient $\langle (a)(c) \rangle$, alors la première occurrence de a dans T doit apparaître avant la dernière occurrence de c. Partant de là, la première occurrence de (b) dans T doit donc apparaître après la première occurrence de a ou avant la dernière occurrence de c.

Étant donnée la définition de classe d'équivalence des séquences, le lemme suivant est immédiat :

Lemme 4.1. Soit $\mathcal{S}$ un ensemble de séquences. $\equiv_{\mathcal{S}}$ est une relation d'équivalence sur l'ensemble de toutes les séquences possibles $\mathcal{T}$. Le nombre de classes d'équivalences $|\mathcal{T} / \equiv_{\mathcal{S}}|$ est au plus $2^{|\mathcal{S}|}$.

Démonstration. Il s'agit d'une conséquence immédiate du fait que pour chaque transaction $T \in \mathcal{T}$, $[T]_{\mathcal{S}} = \mathcal{E}(S \in \mathcal{S} \mid S \preceq T, S \in \mathcal{S} \mid S \not\preceq T)$. □

Sémantiquement, lorsque $\mathcal{E}(\mathcal{IN}, \mathcal{OUT}) = \emptyset$ cela veut dire qu'il est impossible de trouver une séquence $s \in \mathcal{T}$ qui supporte toutes les séquences de l'ensemble $\mathcal{IN}$ et qui ne supporte aucune des

séquences présentes dans $\mathcal{OUT}$. Pour plus de clarté dans le formalisme, les classes d'équivalences peuvent être écrites en utilisant uniquement les éléments maximaux présents dans les ensembles $\mathcal{IN}$ et $\mathcal{OUT}$.

Ici, une première divergence apparaît entre le domaine de l'extraction des itemsets fréquents et celui des motifs séquentiels fréquents. Alors que la structure de classes d'équivalences pour les itemsets est d'une simplicité extrême, celle des motifs séquentiels ne l'est pas du tout. Ainsi pour les itemsets, $\mathcal{E}(\mathcal{IN}, \mathcal{OUT})$ est non-vide si et seulement si, chaque itemset de l'ensemble $\mathcal{OUT}$ possède au moins un item qui n'est pas présent dans un des itemsets contenus dans l'ensemble $\mathcal{IN}$.

Exemple 4.5. $\mathcal{E}(\{ab, ac\}, \{bc\}) = \emptyset$ car chaque transaction qui contient les itemsets ab et ac, doit aussi contenir bc.

Il est évident que décider si une classe d'équivalence pour les itemsets est vide est un exercice trivial qui peut être traité en temps linéaire. Mais qu'en est-il du problème de *décider si une classe d'équivalence pour les motifs séquentiels est vide ?*

Considérons par exemple la classe d'équivalence $\mathcal{E}(\{\langle (a)(b)\rangle, \langle (a)(c)\rangle\}, \{\langle (b)(c)\rangle\})$. Cette classe est non-vide puisqu'elle contient au moins la séquence $\langle (a)(c)(b)\rangle$. Le lemme 4.2 décrit la complexité de ce problème décisionnel. De plus, ce lemme est d'une importance capitale et sera utilisé notamment dans l'étude du problème `SEQFREQSAT`.

Lemme 4.2. Soit $\mathcal{IN}$ et $\mathcal{OUT}$ des ensembles de séquences. Décider si $\mathcal{E}(\mathcal{IN}, \mathcal{OUT})$ est non-vide est **NP**-complet.

Démonstration.

- **NP** :
  Si $\mathcal{E}(\mathcal{IN}, \mathcal{OUT})$ est non-vide, alors l'ensemble contient au moins 1 séquence T de taille au plus $\sum_{S\in\mathcal{IN}} taille(S)$, avec $taille(\langle is_1, \ldots, is_k\rangle) = \sum_{j=1^k} |is_j|$. Supposons maintenant que T est une séquence présente dans $\mathcal{E}(\mathcal{IN}, \mathcal{OUT})$. Alors, pour chaque séquence $S \in \mathcal{IN}$, il existe un ensemble d'indices $i_1 < i_2 < \ldots < i_{|S|}$, tel que le $j^{\text{ème}}$ itemset dans S est un sous-itemset du $i_j{}^{\text{ème}}$ sous-itemset de T. Fixons maintenant pour chaque séquence $S \in \mathcal{IN}$ un des ensembles d'indices $i[S]$ possibles. Construisons maintenant la séquence T' sous-séquence de T : soit $1 \leqslant t_1 < \ldots < t_m \leqslant |T|$ l'ensemble des indices $\cup_{S\in\mathcal{IN}} i[S]$.

$$T' = \left\langle \bigcup_{\substack{S\in\mathcal{IN} \\ t_j=i_k\in i[S]}} S[k] \right\rangle_{j=1}^{m}$$

  Comme $T' \preceq T$, alors pour toute séquence $S \in \mathcal{OUT}$, $S \not\preceq T'$. De plus, T' est construit de telle manière que pour chaque séquence $S \in \mathcal{IN}$, $S \preceq T'$ et $taille(T') \leqslant \sum_{S\in\mathcal{IN}} taille(S)$. La séquence T' représente donc bien un certificat valide vérifiable en temps polynomial pour

montrer que $\mathcal{E}(\mathcal{IN}, \mathcal{OUT})$ est non-vide.

Illustrons cette notion avec un exemple :
Soient $\mathcal{IN} = \{\langle(a)(b)\rangle, \langle(c)(a)\rangle\}$ et $\mathcal{OUT} = \{\langle(b)(a)\rangle\}$. Supposons maintenant que $\mathcal{E}(\mathcal{IN}, \mathcal{OUT})$ est non-vide, alors il existe bien une séquence $T \in \mathcal{E}(\mathcal{IN}, \mathcal{OUT})$ de taille 4 : $\langle(c)(a)(a)(b)\rangle$. Chaque séquence S de l'ensemble $\mathcal{IN}$ est incluse dans T. On peut donc construire les ensembles d'indices suivants : $i[\langle(a)(b)\rangle] = \{2, 4\}$ et $i[\langle(c)(a)\rangle] = \{1, 2\}$. Nous pouvons alors construire la séquence T' en se basant sur l'ensemble des indices $\cup_{S \in \mathcal{IN}} i[S] = \{1, 2, 4\}$. La séquence $T' = \langle(c)(a)(b)\rangle$ représente bel et bien dans ce cas là un certificat qui pourra être vérifié en temps polynomial (en comparant les inclusions sur l'ensemble $\mathcal{OUT}$).

– **NP**-complet :
Considérons le problème décisionnel EQUAL-3COL qui est une variante du problème des 3-colorations 3COL de graphes. Le problème décisionnel EQUAL-3COL peut s'énoncer ainsi : *Soit un graphe G avec $3k$ sommets, est-ce qu'il existe une 3-coloration possible des sommets en utilisant chaque couleur exactement k fois ?* Ce problème est en fait équivalent au problème décisionnel 3COL ; ainsi il est facile de voir qu'un graphe G admet une 3-coloration si et seulement si le graphe associé contenant 3 fois le graphe G est EQUAL-3COL.
Supposons que G contient $3k$ sommets. Nous allons montrer comment procéder à une réduction du problème 3COL avec le graphe G au problème décisionnel $\mathcal{E}(\mathcal{IN}, \mathcal{OUT}) \neq \emptyset$. Soit $V = \{v_1, \ldots, v_{3k}\}$, l'ensemble des sommets et E l'ensemble des arêtes de G. L'ensemble des items est $\mathcal{I} = \{i_1, i_2, \ldots, i_n, i_{n+1}, R, G, B\}$ avec $n = 3k$. Les ensembles de séquences $\mathcal{IN}$ et $\mathcal{OUT}$ seront construits de telle manière que les séquences dans $\mathcal{E}(\mathcal{IN}, \mathcal{OUT})$ soient de la forme $\langle(i_1)(C_1)\ldots(i_n)(C_{3k})(i_{3k+1})\rangle$ avec C_i de valeur R, G ou B et tel que $C(v_i) = C_i$ pour tout $i = 1 \ldots 3k$. Ainsi, chaque séquence présente dans $\mathcal{E}(\mathcal{IN}, \mathcal{OUT})$ est une coloration valide pour le graphe G.
Nous décrivons maintenant le type de séquences qui doivent être présentes dans l'ensemble $\mathcal{IN}$:

1. $\langle(i_1)(i_2)\ldots(i_{3k})(i_{3k+1})\rangle$, *tous les itemsets doivent être présents.*
2. $\left\langle \overbrace{(C)(C)\ldots(C)}^{k\times} \right\rangle$, pour $C = R, G, B$ *chaque couleur apparaît au moins k fois.*

Ces séquences sont nécessaires pour avoir dans $\mathcal{E}(\mathcal{IN}, \mathcal{OUT})$ des séquences de la forme $\langle i_1 C_1 \ldots i_n C_{3k} i_{3k+1}\rangle$ avec C_i de valeur R, G ou B.
Nous décrivons ensuite les séquences qui ne doivent pas apparaître dans l'ensemble $\mathcal{E}(\mathcal{IN}, \mathcal{OUT})$. Ces séquences sont donc nécessairement dans $\mathcal{OUT}$:

1. $\langle(i, j)\rangle$, $\forall i, j \in \mathcal{I}$ et $i \neq j$, *tous les itemsets dans les séquences de $\mathcal{E}(\mathcal{IN}, \mathcal{OUT})$ sont des singletons.*

2. $\langle (i_j)(i_j)\rangle$, $j = 1 \dots 3k+1$, *tous les itemsets des séquences de* $\mathcal{E}(\mathcal{IN}, \mathcal{OUT})$ *apparaissent au plus une fois dans la séquence.*
3. $\left\langle \overbrace{(C)(C)\dots(C)}^{k+1\times} \right\rangle$, pour $C = R, G, B$, *chaque couleur apparaît au plus* k *fois dans les séquences de* $\mathcal{E}(\mathcal{IN}, \mathcal{OUT})$.
4. $\langle (i_j)(C_1)(C_2)(i_{j+1})\rangle$, $j = 1 \dots 3k$, *entre deux itemsets des séquences de* $\mathcal{E}(\mathcal{IN}, \mathcal{OUT})$ *nous avons au plus une couleur avec* $C_1, C_2 = R, G, B$
5. $\langle (i_j)(C)(i_{j+1})(i_k)(C)(i_{k+1})\rangle, \forall (v_j, v_k) \in E$, *deux sommets adjacents ne peuvent pas avoir la même couleur* C avec $C = R, G, B$.

De cette manière, chaque séquence présente dans $\mathcal{E}(\mathcal{IN}, \mathcal{OUT})$ décrit une 3-coloration possible de G. □

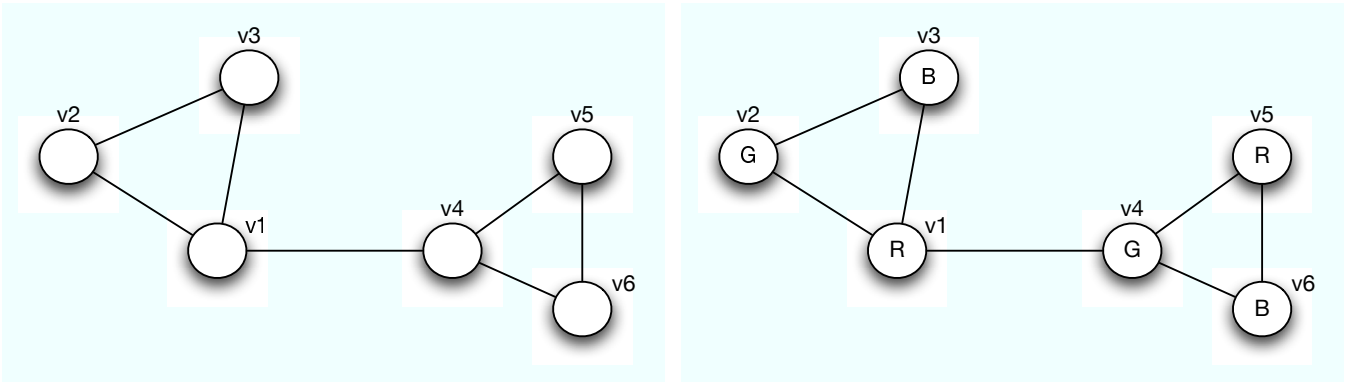
Nous illustrons cette réduction dans l'exemple suivant.

Exemple 4.6. Posons $k = 2$ et soit $G = (V, E)$ un graphe avec $V = \{v_1, v_2, v_3, v_4, v_5, v_6\}$ l'ensemble des sommets (voir Figure 4.3(a)) et soit $\mathcal{I} = \{i_1, i_2, \dots, i_7, R, G, B\}$ l'ensemble des items utilisés dans la construction de nos séquences. Afin que chaque séquence dans $\mathcal{E}(\mathcal{IN}, \mathcal{OUT})$ puisse décrire une 3-coloration possible de G, les ensembles $\mathcal{IN}$ et $\mathcal{OUT}$ doivent être construits de la manière suivante :

$$\mathcal{IN} = \left\{ \begin{array}{l} \langle (i_1)(i_2)(i_3)(i_4)(i_5)(i_6)(i_7)\rangle \\ \langle (R)(R)\rangle \\ \langle (G)(G)\rangle \\ \langle (B)(B)\rangle \end{array} \right\} \quad \text{et} \quad \mathcal{OUT} = \left\{ \begin{array}{l} \langle (i_1, i_2)\rangle \\ \langle (i_1, i_3)\rangle \\ \vdots \\ \langle (B, G)\rangle \\ \langle (i_1)(i_1)\rangle \\ \vdots \\ \langle (i_7)(i_7)\rangle \\ \langle (i_1)(R)(R)(i_2)\rangle \\ \langle (i_1)(R)(G)(i_2)\rangle \\ \vdots \\ \langle (i_6)(G)(G)(i_7)\rangle \end{array} \right\}$$

De cette manière, les séquences dans l'ensemble $\mathcal{E}(\mathcal{IN}, \mathcal{OUT})$ décrivent une 3-coloration possible de G avec chaque couleur apparaissant k fois.

La séquence $\langle (i_1)(R)(i_2)(G)(i_3)(B)(i_4)(G)(i_5)(R)(i_6)(B)(i_7)\rangle$ et sa coloration associée dans le graphe G est illustrée dans la Figure 4.3(b).



(a) Graphe G

(b) Graphe G avec une 3-coloration possible (chaque couleur apparaissant 2 fois)

FIG. 4.3: Graphe G et une 3-coloration possible selon la séquence présentée dans l'exemple 4.6

Les classes d'équivalences pour les motifs séquentiels sont d'un point de vue théorique très intéressantes dans le domaine de l'extraction de motifs. Ces classes d'équivalences permettent ainsi de regrouper et caractériser les séquences selon les sous-séquences qu'elles supportent ou non. Ainsi, plusieurs approches ayant trait à la fouille pourraient bénéficier de ce genre de représentation comme le problème de génération de candidats avec des contraintes (*Nous voulons générer toutes les séquences qui contiennent un ensemble de motifs mais ne supportant pas les motifs X et Y*) ou le problème de génération de bases de données contenant un ensemble précis de motifs [Mie03]. Malheureusement, la complexité du problème rend ce genre d'applications impossible. Dans la prochaine section nous montrerons la connexion directe entre les classes d'équivalences pour les motifs séquentiels et le problème de satisfiabilité de fréquences pour les motifs séquentiels.

5 Le problème de satisfiabilité de fréquence pour les motifs séquentiels : `SEQFREQSAT`

Dans cette section, nous essayons de modéliser les informations sur les fréquences pour un ensemble donné de séquences. Nous nous basons sur l'utilisation de *contraintes de fréquences* comme proposé dans [Cal03]. Les règles de dérivations de supports des séquences que nous pouvons extraire sont de fait issues de ces contraintes.

Le problème `SEQFREQSAT` est un problème de décision. Le problème s'énonce de la manière suivante : *étant donné un ensemble de séquences et leurs fréquences associées, est-ce qu'il existe une base de données respectant ces contraintes ?* Nous allons illustrer dans l'exemple suivant l'intérêt que peut avoir ce problème dans la problématique d'extraction de motifs séquentiels et plus précisément dans l'étape de comptage de fréquences (ou de supports).

Exemple 4.7. Soit l'ensemble de contraintes de fréquences sur les séquences

$$\mathcal{CS} = \left\{ \begin{array}{l} f_{\langle (a) \rangle} = \frac{1}{2} \\ f_{\langle (b) \rangle} = 1 \\ f_{\langle (a)(b) \rangle} = \frac{1}{2} \\ f_{\langle (a,b) \rangle} = \frac{1}{2} \end{array} \right\}$$

Cette instance du problème SEQFREQSAT est satisfiable puisque la base de données transactionnelles $\mathcal{D}$ suivante respecte les contraintes de $\mathcal{CS}$.

$\mathcal{D} =$

Id. Séquences	Séquences
S_1	(a)(a,b)
S_2	(b)

5.1 Définitions

Nous rappelons et étendons quelques définitions sur les contraintes de fréquences issues de [CG02a, Cal03, Cal08].

Définition 4.10 (Contrainte de fréquence)**.** Une contrainte de fréquence pour une séquence s est définie telle que : $f_s \in [l, u]$ où $l, u \in [0, 1]$.

Définition 4.11 (Satisfiabilité)**.** Une base de données transactionnelles $\mathcal{D}$ *satisfait* une contrainte de fréquence sur la séquence s, $f_s \in [l, u]$ si $l \leqslant f_s \leqslant u$. Une base de données transactionnelles $\mathcal{D}$ *satisfait totalement* un ensemble de contraintes de fréquences $\mathcal{CS}$ si et seulement si chaque contrainte de fréquence dans $\mathcal{CS}$ est *satisfaite* par $\mathcal{D}$.

Définition 4.12 (Implication de contrainte de fréquence)**.** Un ensemble de contraintes de fréquences $\mathcal{CS}$ *implique une contrainte de fréquence* sur la séquence s, $f_s \in [l, u]$, si pour chaque base de données transactionnelles $\mathcal{D}$ satisfaisant $\mathcal{CS}$ nous avons $f_s \in [l, u]$.

Avec ces définitions, nous pouvons définir formellement le problème SEQFREQSAT qui nous permettra d'étudier les implications sur les contraintes de fréquences pour les séquences :

Définition 4.13 (Problème SEQFREQSAT)**.** Étant donné un ensemble $\mathcal{CS}$ de contraintes de fréquences pour les séquences, le problème SEQFREQSAT consiste à décider s'il existe une base de données $\mathcal{D}$ telle que $\mathcal{D}$ satisfait $\mathcal{CS}$.

5.2 Implications sur les contraintes de fréquences

Afin d'étudier les implications sur les contraintes de fréquences dans le cadre des itemsets, T. Calders a introduit dans [Cal03], la notion importante de *fraction*. Pour un itemset I, une I-fraction, notée $\mathcal{F}_I$, est la fraction de transactions contenant exactement I comme ensemble d'items.

Cette définition permet ainsi dans le cadre des itemsets de reformuler la définition de fréquence d'un itemset I :

$$f_I = \sum_{I \subseteq J \subseteq \mathcal{I}} \mathcal{F}_J \tag{4.4}$$

Nous illustrons ces différentes définitions et concepts dans l'exemple suivant :

Exemple 4.8. Soit la base de données suivante avec $\mathcal{I} = \{a, b, c, d\}$:

$\mathcal{D}_I =$

ID Transactions	Items
T_1	$\{a, b, c\}$
T_2	$\{b, c\}$
T_3	$\{b, c, d\}$

La fraction de l'itemset (b, c), noté $\mathcal{F}_{(b,c)}$, est ici égale à $\frac{1}{3}$, puisque seule la transaction T_2 contient exactement l'itemset (b, c) comme ensemble d'items. De même, $\mathcal{F}_{(a,c)} = 0$.

Selon la nouvelle formulation de la fréquence, nous avons :

- $f_{(b,c)} = \mathcal{F}_{(b,c)} + \mathcal{F}_{(a,b,c)} + \mathcal{F}_{(b,c,d)} = \frac{1}{3} + \frac{1}{3} + \frac{1}{3} = 1$
- $f_{(a,c)} = \mathcal{F}_{(a,c)} + \mathcal{F}_{(a,b,c)} + \mathcal{F}_{(a,c,d)} = 0 + \frac{1}{3} + 0 = \frac{1}{3}$

Un des intérêts de ces nouvelles définitions et formulations en se basant sur le concept de fraction est de pouvoir faire le lien entre un ensemble de contraintes de fréquences pour itemsets $\mathcal{CS}_I$ et un système d'équations linéaires.

Ainsi, dans ses travaux [CG02a, Cal03, Cal08], l'auteur exhibe un lien prouvant qu'un ensemble de contraintes de fréquences pour un ensemble de sous-itemsets d'un itemset I est satisfiable par une base de données $\mathcal{D}$ si et seulement si le système d'équations $Sys(\mathcal{CS}_I)$ suivant, où nous associons pour chaque variable X_J la fraction $\mathcal{F}_J$, possède une solution positive :

$$\mathcal{P}(\mathcal{CS}_I) = \begin{cases} X_J \geqslant 0 & \forall J \subseteq I \\ \sum_{\emptyset \subseteq J \subseteq I} X_J = 1 & \\ \sum_{J \subseteq K \subseteq I} X_K = f_J & \forall J \subseteq I \end{cases}$$

Exemple 4.9. Soit $\mathcal{I} = \{a, b, c\}$ et soit l'ensemble de contraintes de fréquences sur les itemsets

$$\mathcal{CS}_I = \left\{ \begin{array}{l} f_{\emptyset} = 1 \\ f_{(a)} = \frac{2}{3} \\ f_{(b)} = \frac{2}{3} \\ f_{(c)} = \frac{2}{3} \\ f_{(a,b)} = \frac{1}{3} \\ f_{(a,c)} = \frac{1}{3} \\ f_{(b,c)} = \frac{1}{3} \\ f_{(a,b,c)} = 0 \end{array} \right\}$$

Cet ensemble de contraintes de fréquences est satisfiable si et seulement si le système d'équations suivant à une solution positive pour tous les X_J :

$$Sys(\mathcal{CS}_I) = \left\{ \begin{array}{ll} 1 & = X_{\emptyset} + X_{(a)} + X_{(b)} + X_{(c)} + X_{(a,b)} + X_{(a,c)} + X_{(b,c)} + X_{(a,b,c)} \\ f_{(a)} & = X_{(a)} + X_{(a,b)} + X_{(a,c)} + X_{(a,b,c)} \\ f_{(b)} & = X_{(b)} + X_{(a,b)} + X_{(b,c)} + X_{(a,b,c)} \\ f_{(c)} & = X_{(c)} + X_{(a,c)} + X_{(b,c)} + X_{(a,b,c)} \\ f_{(a,b)} & = X_{(a,b)} + X_{(a,b,c)} \\ f_{(b,c)} & = X_{(b,c)} + X_{(a,b,c)} \\ f_{(a,c)} & = X_{(a,c)} + X_{(a,b,c)} \\ f_{(a,b,c)} & = X_{(a,b,c)} \end{array} \right.$$

Ce système contient 8 équations et 8 variables. La solution du système est :

$$\{X_{\emptyset} = 0; X_{(a)} = 0; X_{(b)} = 0; X_{(c)} = 0; X_{(a,b)} = \frac{1}{3}; X_{(a,c)} = \frac{1}{3}; X_{(b,c)} = \frac{1}{3}; X_{(a,b,c)} = 0\}$$

Les variables X_J étant associées aux fractions, il est possible alors d'exhiber une base de données $\mathcal{D}_I$ qui aura trois transactions contenant les itemsets suivants : (a, b), (a, c) et (b, c).

$\mathcal{D}_I =$

ID Transactions	Items
T_1	$\{a, b\}$
T_2	$\{b, c\}$
T_3	$\{a, c\}$

Cette base de données satisfait bien l'ensemble de contraintes de fréquences pour les itemsets.

Cette méthode très efficace est malheureusement impossible dans le cadre des séquences. Il n'est donc pas possible de répondre à un problème SEQFREQSAT en utilisant un concept de *fraction* pour les séquences. La proposition et l'exemple suivants illustrent l'impossibilité d'appliquer directement cette méthode dans le cadre des motifs séquentiels.

Proposition 4.1. Supposons que pour une séquence s, nous ayons une définition similaire que le concept de fraction pour les itemsets. Notons cette fraction pour une séquence s la s-fraction, notée $\mathcal{SF}_s$. $\mathcal{SF}_s$ est la fraction de transactions contenant exactement s comme séquence de données et supposons que comme pour les itemsets :

$$f_s = \sum_{s \preceq s' \preceq \mathcal{T}(\mathcal{I})} \mathcal{SF}_{s'}$$

Supposons, de plus, qu'un ensemble de contraintes de fréquences de séquences $\mathcal{CS}$ pour les sous-séquences d'une séquence s est satisfiable par une base de données transactionnelles $\mathcal{D}$ si et seulement si le système d'équations $Sys(\mathcal{CS})$ suivant, où nous associons pour chaque variable $X_{s'}$ la s-fraction $\mathcal{SF}_{s'}$, possède une solution positive :

$$Sys(\mathcal{CS}) = \begin{cases} X_{s'} \geqslant 0 & \forall s' \preceq s \\ \sum_{\langle () \rangle \preceq s' \preceq s} X_{s'} = 1 & \\ \sum_{s' \preceq k \preceq s} X_k = f_{s'} & \forall s' \preceq s \end{cases}$$

Exemple 4.10. Soit $\mathcal{I} = \{a, b\}$ et supposons l'ensemble de contraintes de fréquences de séquences suivant :

$$\mathcal{CS} = \left\{ \begin{array}{l} f_{\langle () \rangle} = 1 \\ f_{\langle (a) \rangle} = 1 \\ f_{\langle (b) \rangle} = 1 \\ f_{\langle (a)(b) \rangle} = 0 \end{array} \right\}$$

D'après la proposition 4.1, l'ensemble $\mathcal{CS}$ est satisfiable si et seulement si le système d'équations suivant possède une solution positive :

$$Sys(\mathcal{CS}) = \begin{cases} 1 & = X_{\langle () \rangle} + X_{\langle (a) \rangle} + X_{\langle (b) \rangle} + X_{\langle (a)(b) \rangle} \\ 1 & = X_{\langle (a) \rangle} + X_{\langle (a)(b) \rangle} \\ 1 & = X_{\langle (b) \rangle} + X_{\langle (a)(b) \rangle} \\ 0 & = X_{\langle (a)(b) \rangle} \end{cases}$$

Ce système ne possède pas de solutions positives puisque les équations impliquent que : $X_{\langle (a)(b) \rangle} = 0$ donc $X_{\langle (a) \rangle} = 1$ et $X_{\langle (b) \rangle} = 1$ ce qui implique finalement $X_{\langle () \rangle} = -1$. D'où contradiction car il existe bien au moins une base de données transactionnelles satisfaisant l'ensemble de contraintes de fréquences $\mathcal{CS}$:

$\mathcal{D} =$

Id. Séquences	Séquences
S_1	(b)(a)

L'exemple précédent montre bien que la proposition 4.1 est incomplète puisqu'il faut prendre en compte dans le cas des motifs séquentiels toutes les super-séquences ayant une action sur la

fréquence d'une séquence s (i.e. arriver à regrouper les séquences selon ce qu'elles supportent comme sous-séquences). Pour cela, nous utilisons les classes d'équivalences afin de résoudre ce problème.

Définition 4.14. Soit $\mathcal{CS}$ un ensemble de contraintes de fréquences pour les séquences et soit $\mathcal{S}$ l'ensemble de séquences présentes dans $\mathcal{CS}$. $Sys(\mathcal{CS})$ représente le système d'équations suivant : nous associons la variable $X_{\mathcal{S}'}$ pour chaque $\mathcal{S}' \subseteq \mathcal{S}$ tel que $\mathcal{E}(\mathcal{S}', \mathcal{S}) \neq \emptyset$.

$$Sys(\mathcal{CS}) = \begin{cases} \sum\limits_{\substack{\{S_i\} \subseteq \mathcal{S}' \subseteq \mathcal{S} \\ \mathcal{E}(\mathcal{S}', \mathcal{S}) \neq \emptyset}} X_{\mathcal{S}'} = f_i & \forall S_i \in \mathcal{S} \\ \\ \sum\limits_{\substack{\mathcal{S}' \subseteq \mathcal{S} \\ \mathcal{E}(\mathcal{S}', \mathcal{S}) \neq \emptyset}} X_{\mathcal{S}'} = 1 \end{cases} \tag{4.5}$$

Lemme 4.3. Soit $\mathcal{CS}$ un ensemble de contraintes de fréquences pour les séquences. $\mathcal{CS}$ est dans SEQFREQSAT (i.e. $\mathcal{CS}$ est satisfiable) si et seulement si le système d'équations $Sys(\mathcal{CS})$ possède une solution positive.

Démonstration.

Si. Supposons que que le système d'équations $Sys(\mathcal{CS})$ possède une solution positive pour chaque $X_{\mathcal{S}'} = a_{\mathcal{S}'}$. Posons d le plus petit commun multiple des dénominateurs des $a_{\mathcal{S}'}$, alors nous pouvons exhiber une base de données transactionnelles $\mathcal{D}$ contenant d transactions telles que $\mathcal{D}$ satisfait $\mathcal{CS}$. Il suffit pour cela de fixer pour chaque variable $X_{\mathcal{S}'}$ une séquence $T_{\mathcal{S}'} \in \mathcal{E}(\mathcal{S}', \mathcal{S})$. De cette manière, $\mathcal{D}$ contient $d \times a_{\mathcal{S}'}$ pour chaque variable $X_{\mathcal{S}'}$ dans le système $Sys(\mathcal{CS})$.

Seulement si. Soit $\mathcal{D}$ une base de données transactionnelles satisfaisant l'ensemble de contraintes de fréquences $\mathcal{CS}$. Pour chaque variable $X_{\mathcal{S}'}$ du système $Sys(\mathcal{CS})$ fixons :

$$X_{\mathcal{S}'} = \frac{|\mathcal{D} \cap \mathcal{E}(\mathcal{S}', \mathcal{S})|}{|\mathcal{D}|}$$

De cette manière les variables $X_{\mathcal{S}'}$ représentent une solution positive au système $Sys(\mathcal{CS})$. □

Exemple 4.11. Reprenons l'ensemble de contraintes de fréquences pour les séquences présentées dans l'exemple 4.10.

$$\mathcal{CS} = \left\{ \begin{array}{l} f_{\langle () \rangle} = 1 \\ f_{\langle (a) \rangle} = 1 \\ f_{\langle (b) \rangle} = 1 \\ f_{\langle (a)(b) \rangle} = 0 \end{array} \right\}$$

Soit $\mathcal{S}$ l'ensemble de séquences présentes dans $\mathcal{CS}$:

$$\mathcal{S} = \left\{ \begin{array}{l} \langle () \rangle \\ \langle (a) \rangle \\ \langle (b) \rangle \\ \langle (a)(b) \rangle \end{array} \right\}$$

Les ensembles $\mathcal{S}'$ pour lesquels $\mathcal{E}(\mathcal{S}', \mathcal{S}) \neq \emptyset$ sont :

$$\begin{array}{rcl} \mathcal{S}_1 & = & \{\langle () \rangle\} \\ \mathcal{S}_2 & = & \{\langle () \rangle, \langle (a) \rangle\} \\ \mathcal{S}_3 & = & \{\langle () \rangle, \langle (b) \rangle\} \\ \mathcal{S}_4 & = & \{\langle () \rangle, \langle (a) \rangle, \langle (b) \rangle\} \\ \mathcal{S}_5 & = & \{\langle () \rangle, \langle (a) \rangle, \langle (b) \rangle, \langle (a)(b) \rangle\} \end{array}$$

Ainsi, $\mathcal{CS}$ est satisfiable si et seulement si il existe une solution positive au système d'équations suivant :

$$\left\{ \begin{array}{rcl} X_1 + X_2 + X_3 + X_4 + X_5 & = & 1 \\ X_2 + X_4 + X_5 & = & 1 \\ X_3 + X_4 + X_5 & = & 1 \\ X_5 & = & 0 \end{array} \right.$$

Ce système possède une solution :

$$\{X_1 = 0, X_2 = 0, X_3 = 0, X_4 = 1, X_5 = 0\}$$

Il est donc possible d'exhiber une base de données transactionnelles $\mathcal{D}$ contenant une seule séquence.

Cette séquence est choisie dans la classe d'équivalence $\mathcal{E}(\mathcal{S}_4, \mathcal{S}) = \{\langle (a,b) \rangle, \langle (b)(a) \rangle, \langle (b)(a)(a) \rangle \ldots\}$. Voici un exemple de bases satisfaisant l'ensemble de contraintes $\mathcal{CS}$:

$\mathcal{D}_1 =$

Id. Séquences	Séquences
S_1	(b)(a)

$\mathcal{D}_2 =$

Id. Séquences	Séquences
S_1	(a,b)
S_1	(b)(b)(a)

5.3 Complexité

Nous étudions la complexité du problème `SEQFREQSAT`. Ce problème, comme dans le cas du problème similaire associé aux itemsets (`FREQSAT`), est **NP**-complet. L'inclusion dans la classe de complexité **NP** a déjà été implicitement abordé puisque nous réduisons tout problème `SEQFREQSAT` à un problème de programmation linéaire (cf. Théorème 4.14 et Lemme 4.3) ayant pour une séquence maximale de taille l, $2^l + z$ variables (z étant le nombre de classes d'équivalences construits sur plusieurs séquences) et 2^l équations linéaires.

Lemme 4.4 ([Chv83])**.** Si un système de n équations linéaires possède une solution positive alors ce système possède aussi une solution avec au plus n variables positives.

Ainsi, si le programme linéaire (le système $Sys(\mathcal{CS})$) avec $2^l + z$ variables possède une solution, alors l'ensemble de variables strictement positives (au plus $2^l + z$ variables) représente un certificat vérifiable [Cal03, Pap94]. De plus, il faut noter qu'il faut présenter pour chacune de ces variables représentant une classe d'équivalence non-vide un certificat : en l'occurrence une séquence appartenant à la classe d'équivalence.

Théorème 4.1. `SEQFREQSAT` est **NP**-complet.

Démonstration. La preuve que `SEQFREQSAT` est **NP**-difficile dérive directement de la complexité de décider si une classe d'équivalence est non-vide (Lemme 4.2) et du Lemme 4.3. □

6 Cas spécial

La **NP**-complétude du problème `SEQFREQSAT` nous pousse à étudier un cas spécial du problème de satisfiabilité d'un ensemble de contraintes sur les fréquences de séquences. Dans cette section, nous étudions le cas le plus intéressant puisqu'il permet l'extraction de règles de dérivations pour la fréquence d'une séquence.

Comme nous l'avons souligné dans la section précédente, le problème `SEQFREQSAT` est intimement lié au problème de décider si une classe d'équivalence $\mathcal{E}(\mathcal{S}', \mathcal{S})$ est non-vide. Nous commençons par exhiber un ensemble de propriétés permettant de caractériser des classes d'équivalences vides de manière rapide dans le cas où les ensembles de séquences $\mathcal{IN}$ et $\mathcal{OUT}$ sont des hyper-treillis complets.

Propriété 4.1. Soit $\mathcal{S}$ un ensemble de séquences tel que cet ensemble muni de la relation $\preceq$ forme un hyper-treillis contenant une borne supérieure (i.e. une séquence maximale) et une borne inférieure (la séquence $\langle()\rangle$).

Soit $\mathcal{S}' \subseteq \mathcal{S}$, si la borne inférieure de $\mathcal{S}'$ n'est pas la séquence $\langle()\rangle$ alors $\mathcal{E}(\mathcal{S}', \mathcal{S}) = \emptyset$.

Démonstration. Trivial, puisque pour chaque séquence $s \in \mathcal{T}$, $\langle()\rangle \preceq s$.

□

Propriété 4.2. Si l'ensemble $\mathcal{S}'$ muni de la relation $\preceq$ n'est pas un hyper-treillis complet alors $\mathcal{E}(\mathcal{S}', \mathcal{S}) = \emptyset$.

Démonstration. Soit $\mathcal{E}(\mathcal{S}', \mathcal{S}) = \{T\}$ et $s \in \mathcal{S}'$ avec $s \preceq T$. Alors chaque sous-séquence de s est aussi incluse dans T et doit donc être contenue dans$\mathcal{S}'$, par construction, $\mathcal{S}'$ forme alors un hyper-treillis complet.

□

Exemple 4.12. Soit $\mathcal{I} = \{a, b\}$, et

$$\mathcal{S} = \left\{ \begin{array}{l} \langle(ab)(ab)\rangle \\ \langle(ab)(a)\rangle \\ \langle(ab)(b)\rangle \\ \langle(a)(ab)\rangle \\ \langle(b)(ab)\rangle \\ \langle(a)(a)\rangle \\ \langle(a)(b)\rangle \\ \langle(ab)\rangle \\ \langle(b)(a)\rangle \\ \langle(b)(b)\rangle \\ \langle(a)\rangle \\ \langle(b)\rangle \\ \langle\{\}\rangle \end{array} \right\} \qquad \mathcal{S}' = \left\{ \begin{array}{l} \langle(ab)(b)\rangle \\ \langle(ab)(a)\rangle \\ \langle(ab)\rangle \\ \langle(a)(a)\rangle \\ \langle(a)(b)\rangle \\ \langle(a)\rangle \\ \langle(b)\rangle \\ \langle\{\}\rangle \end{array} \right\}$$

$\mathcal{E}(\mathcal{S}', \mathcal{S}) = \emptyset$ car l'ensemble $\mathcal{S}$ ne forme pas un hyper-treillis complet : la séquence $\langle(ab)(a)\rangle$ supporte nécessairement la séquence $\langle(b)(a)\rangle$, or celle-ci n'est pas présente dans l'ensemble $\mathcal{S}'$. Le même raisonnement peut être tenu pour la séquence $\langle(ab)(b)\rangle$ et la séquence $\langle(b)(b)\rangle$.

Propriété 4.3. Si l'ensemble $\mathcal{S}'$ muni de la relation $\preceq$ forme un hyper-treillis complet avec une unique borne supérieure (une séquence maximale unique) alors $\mathcal{E}(\mathcal{S}', \mathcal{S}) \neq \emptyset$.

Démonstration. Trivial, soit T la séquence maximale unique dans l'ensemble $\mathcal{S}'$. Selon la définition des séquences, $T \preceq T$ et donc, T est aussi contenu dans $\mathcal{E}(\mathcal{S}', \mathcal{S})$ (c'est d'ailleurs l'élément minimal). □

Propriété 4.4. Soit T l'unique borne supérieure de l'ensemble $\mathcal{S}$, et soit $\mathcal{S}' = \mathcal{S} \setminus T$, alors $\mathcal{E}(\mathcal{S}', \mathcal{S}) \neq \emptyset$ (i.e. pour chaque séquence s, il existe une séquence qui supporte toutes les sous-séquences de s mais pas s).

Démonstration. Afin de prouver la propriété, nous exhibons un algorithme récursif permettant de construire une séquence $\mathcal{T}_s$ supportant toutes les sous-séquences de s mais pas s. Soit $s = \langle(it_1)s'\rangle$ où (it_1) représente le premier itemset de taille k de la séquence et s' représente le reste des itemsets (i.e. $(it_2)\ldots(it_n)$).

La séquence $\mathcal{T}_s$ peut être construite de la manière suivante :

$$\mathcal{T}_s = \langle(it_1 - i_1)(it_1 - i_2)....(it1 - i_k)(s')(it_1)(\mathcal{T}_{s'})\rangle$$

L'algorithme fonctionne de manière récursive en éclatant le premier itemset en de multiples nouveaux itemsets. Puis un appel récursif de la fonction est appliqué sur le reste de la séquence $s : s'$. Les deux conditions d'arrêts sont :

1. Si $s = (it_1)$ alors $\mathcal{T} = \langle (it_1 - i_1)(it_1 - i_2) \dots (it_1 - i_k) \rangle$
2. Si $s = \langle (i) \rangle$, une séquence contenant un seul itemset avec un unique item alors arrêter.

La complétude de cet algorithme est prouvée par un raisonnement par récurrence :

Prenons pour cette démonstration le cas le plus simple où tous les itemsets d'une séquence s sont différents et de taille d'itemsets 1 (le cas général est légèrement plus difficile mais suit les mêmes lignes de cette démonstrations). Notons N_s, l'ensemble des sous-séquences strictes de s.

Soit l'hypothèse suivante que nous voulons prouver :

$$\mathcal{P}(n) = \begin{cases} \mathcal{T}_s \text{ est une séquence qui supporte toutes les séquences de } N_s \\ \mathcal{T}_s \text{ ne supporte pas la séquence } s \text{ de taille n} \end{cases}$$

Pour une séquence de taille 2, cette propriété est vérifié (la séquence vérifiant la propriété est : $\langle (a_2)(a_1) \rangle$).

Supposons que la propriété $\mathcal{P}(n)$ est vraie pour pour une séquence s de taille n.

Montrons que cette propriété est vraie pour une séquence de taille $n+1$: $s = \langle (a_1) \dots (a_{n+1}) \rangle$.

Par construction nous savons que $\mathcal{T}_s = \langle (a_2) \dots (a_{n+1})(a_1)\mathcal{T}_{s'} \rangle$ avec $s' = \langle (a_2) \dots (a_{n+1}) \rangle$. Par l'hypothèse de récurrence nous savons que :

1. $\mathcal{T}_{s'}$ supporte les séquences de l'ensemble $N_{s'}$. (1)
2. $\mathcal{T}_{s'}$ ne supporte pas la séquence s'. (2)

Par définition, on sait que $\mathcal{T}_{s'} \preceq \mathcal{T}_s$. Par la relation (1), on sait que pour toute séquence $seq \in N_{s'}$, $seq \preceq \mathcal{T}_{s'}$. Par transitivité, on alors que pour toute séquence $seq \in N_{s'}$, $seq \preceq \mathcal{T}_s$. La première partie de la propriété $\mathcal{P}$ est prouvée.

Par l'absurde. Supposons maintenant que $s \preceq \mathcal{T}_s$. Par construction de $\mathcal{T}_s$, on sait que (a_1) apparaît une seule fois. La seule façon pour que cette contrainte d'apparition soit respectée est que $s' \preceq \mathcal{T}_{s'}$. D'où contradiction avec l'hypothèse de récurrence : $\mathcal{T}_{s'}$ ne supporte pas la séquence s' . Donc la deuxième partie de la propriété est prouvée.

Alors pour toute séquence de taille $n \geqslant 2$, la propriété $\mathcal{P}(n)$ est vraie.

□

Exemple 4.13. Soit la séquence $T = \langle(a)(b)(c)\rangle$, l'ensemble de séquence $\mathcal{S}' = \left\{ \begin{array}{l} \langle(a)(b)\rangle \\ \langle(a)(c)\rangle \\ \langle(b)(c)\rangle \\ \langle(c)\rangle \\ \langle(b)\rangle \\ \langle(a)\rangle \\ \langle()\rangle \end{array} \right\}$ et $\mathcal{S} = \{\langle(a)(b)(c)\rangle\} \cup \mathcal{S}'$.

En utilisant l'algorithme récursif nous avons :

$$\begin{aligned} \mathcal{T}_s &= \left\langle (b)(c)(a)\mathcal{T}_{(b)(c)} \right\rangle \\ &= \left\langle (b)(c)(a)(c)(b)\mathcal{T}_{(c)} \right\rangle \\ &= \langle (b)(c)(a)(c)(b) \rangle \end{aligned}$$

Ainsi, $\mathcal{E}(\mathcal{S}', \mathcal{S}) \neq \emptyset$ puisque la classe d'équivalence contient au moins la séquence $\mathcal{T}_s$ qui supporte toutes les séquences de $\mathcal{S}'$ mais pas la séquence $\langle(a)(b)(c)\rangle$.

Ces propriétés permettent de réaliser que le fait de se ramener à des ensembles de séquences ordonnées selon la relation d'inclusion $\preceq$ et formant un hyper-treillis complet décroît la complexité du problème `SEQFREQSAT`. De plus, ces propriétés ouvrent la voie vers un algorithme permettant d'énumérer toutes les classes d'équivalences $\mathcal{E}(\mathcal{S}', \mathcal{S})$ non-vides pour ce cas spécial.

6.1 Un algorithme d'énumération des classes d'équivalences non-vides

Soit l'ensemble de séquences $\mathcal{S}'$ muni de la relation $\preceq$ formant un hyper-treillis complet avec une unique borne supérieure et une unique borne inférieure ($\{s \mid L \preceq s \preceq U\}$) et soit T la séquence minimale dans la classe d'équivalence (i.e. aucun item de la séquence T ne peut être enlevé sans perdre le support de quelques sous-séquences). Soit T' la séquence générée après avoir enlevé le dernier item de la séquence T, cette séquence est aussi la séquence minimale d'un ensemble de séquence $\mathcal{S}''$:

$$\mathcal{S}' = \mathcal{S}'' \cup \{s \diamond \alpha \mid s \in \mathcal{S}'' \text{ et } s \diamond \alpha \preceq U\} \tag{4.6}$$

Où $s \diamond \alpha$ représente la S-extension ou la I-extension de la séquence s avec l'item $\alpha \in \mathcal{I}(U)$ (i.e, $\langle(a)(a)\rangle \diamond (b) = \langle(a)(a)(b)\rangle$ ou $\langle(a)(a,b)\rangle$). Ainsi, supposons que la séquence s' est dans $\mathcal{S}'$ mais n'est pas présente dans $\mathcal{S}''$. Alors, le dernier item dans la séquence s' doit forcément être l'item α car sinon on aurait $s' \preceq T'$. Donc, si nous posons $s' = s'' \diamond \alpha$, la séquence s'' doit être dans $\mathcal{S}''$ car $s'' \preceq T'$. Cette déduction permet d'exhiber la propriété suivante :

Propriété 4.5. Un ensemble $\mathcal{S}'$ tel que $\mathcal{E}(\mathcal{S}', \mathcal{S}) \neq \emptyset$ peut être formé en étendant un sous-ensemble $\mathcal{S}''$ tel que $\mathcal{E}(\mathcal{S}'', \mathcal{S}) \neq \emptyset$ en procédant à une extension de séquence avec l'item $\alpha \in \mathcal{I}(U)$.

En se basant sur cette propriété, nous pouvons construire un algorithme polynomial en la sortie permettant d'extraire toutes les classes d'équivalences non-vides. L'algorithme est présenté dans les Algorithmes 2 et 3

Algorithme 2 : Algorithme d'énumérations de classes d'équivalences non-vides

Data : La séquence maximale U

Result : L'ensemble $\mathcal{Z}$ contenant toutes les classes d'équivalences non-vides

```
1  L ← ∅;
2  foreach α_u ∈ I(U) do
3      B(⟨{}⟩, U, α_u, Z)
4  return L;
```

Algorithme 3 : Fonction récursive

Data : U la séquence maximale de $\mathcal{S}$; $\mathcal{S}'$; un item α ; l'ensemble des classes d'équivalences non-vides $\mathcal{Z}$

```
1  S'' ← S';
2  foreach s ∈ S' do
3      x ← e ⋄ α;
4      if x ⪯ U then
5          // On rajoute la séquence x à l'ensemble S''.
6          S'' ← x;
7  if |S''| ⩾ |S'| then
8      // S'' est une classe d'équivalence non-vide.
9      // On rajoute l'ensemble S'' à Z
10     L ← S'';
11     //Appel récursif
12     foreach α_u ∈ I(U) do
13         B(S'', U, α_u, Z)
14 return;
```

Nous illustrons cet algorithme dans l'exemple suivant.

Exemple 4.14. Soit l'ensemble de séquences $\mathcal{S} = \left\{ \begin{array}{l} \langle (a)(b) \rangle \\ \langle (a) \rangle \\ \langle (b) \rangle \\ \langle () \rangle \end{array} \right\}$

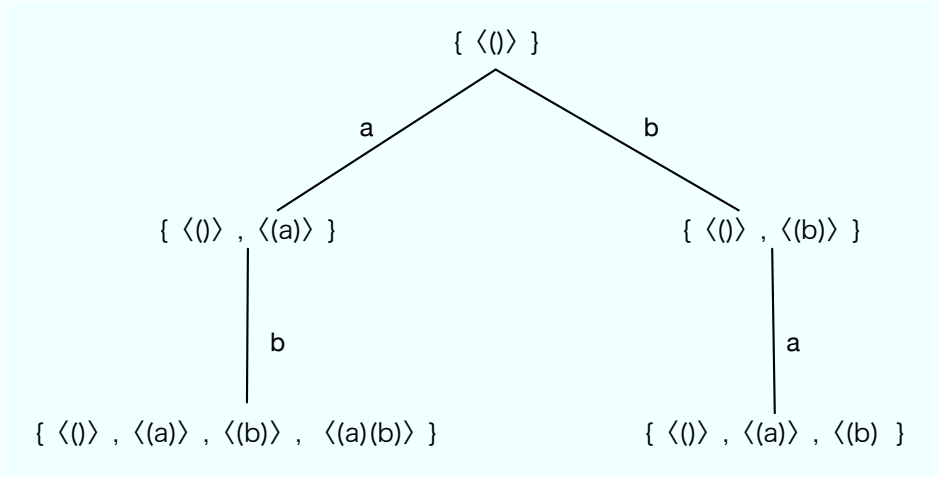


FIG. 4.4: Trace de l'algorithme d'énumération de classes d'équivalences non-vides

avec $U = \langle(a)(b)\rangle$. L'algorithme commence avec l'ensemble $\mathcal{S}' = \{\langle()\rangle\}$. La Figure 4.14 présente une trace de cet algorithme (pour plus de clarté, seules les classes d'équivalences non-vides sont représentées), chaque nœud de l'arbre représente un ensemble de séquences $\mathcal{S}'$ tel que $\mathcal{E}(\mathcal{S}', \mathcal{S}) \neq \emptyset$

Cet algorithme nous montre bien qu'il est possible, dans la pratique, d'énumérer les classes d'équivalences non-vides dans le cas où l'ensemble de séquences $\mathcal{S}$ contient une unique séquence maximale U et une unique séquence minimale (i.e. la séquence $\langle()\rangle$).

Ce cas spécial est en fait retrouvé dans l'algorithme générique basé sur le paradigme *générer-élaguer* : ainsi l'ensemble $\mathcal{S}$ peut être vu comme l'union d'un sous-ensemble de $FSeqs(\mathcal{D}, \sigma)$ contenant toutes les sous-séquences de la séquence nouvellement générée U et la séquence U elle-même. Une question se pose alors : *le problème* `SEQFREQSAT` *peut-il nous aider à dériver des informations plus précises sur la fréquence d'une séquence que celles actuellement utilisées (i.e. l'antimonotonie du support) ?* La réponse à cette question est donnée dans les deux prochaines sous-sections.

6.2 Borne inférieure pour la fréquence d'une séquence

Est-il possible, ayant à priori, un ensemble d'informations sur les fréquences des sous-séquences s de la séquence U de dériver une règle exhibant une borne inférieure pour la fréquence d'une séquence (i.e. $r \leqslant f_U$ avec $r \in \mathbb{R}^+$) sans passer par l'étape de comptage ?

Théorème 4.2. Soit s une séquence et $\mathcal{CS}$ un ensemble de contraintes de fréquences pour toutes les séquences s' telles que $s' \prec s$. S'il existe une base de données transactionnelles $\mathcal{D}$ satisfaisant cet ensemble de contraintes alors la borne inférieure pour la fréquence de la séquence s, notée $LB(S)$, est égale à 0.

Démonstration. La preuve se base sur la propriété 4.4 qui stipule que pour une séquence s, la classe d'équivalence $\mathcal{E}(\{s' \mid s' \prec s\}, \{s' \mid s' \preceq s\})$ est toujours non-vide.

Soit $\mathcal{D}$ la base de données transactionnelles satisfaisant l'ensemble de contraintes $\mathcal{CS}$. Comme $\mathcal{E}(\{s' \mid s' \prec s\}, \{s' \mid s' \preceq s\})$ est non-vide, nous pouvons remplacer chaque séquence dans les transactions de $\mathcal{D}$ qui supporte la séquence s par une séquence issue de la classe d'équivalence $\mathcal{E}(\{s' \mid s' \prec s\}, \{s' \mid s' \preceq s\})$. Cette transformation n'affecte pas la fréquence des sous-séquences de s et donc la nouvelle base satisfait toujours $\mathcal{CS}$. La fréquence de la séquence s dans la nouvelle base de données transactionnelles est quand à elle égale à 0 d'où $LB(S) = 0$. □

Exemple 4.15. Soit $\mathcal{D}$ une base de données transactionnelles contenant la séquence $s = \langle(a)(b)(c)\rangle$. Nous pouvons toujours construire une base de données $\mathcal{D}'$ qui respecte les fréquences de chaque sous-séquence (au sens strict) de la séquence s mais qui réduit la fréquence de s à 0.

$\mathcal{D} =$

Id. Séquences	Séquences
S_1	$(a)(b)(c)$
S_2	$(a)(b)(c)$
S_3	$(c)(a)$
S_4	$(b)(c)$

$\mathcal{D}' =$

Id. Séquences	Séquences
S_1	$(b)(c)(a)(c)(a)(b)$
S_2	$(b)(c)(a)(c)(a)(b)$
S_3	$(c)(a)$
S_4	$(b)(c)$

Corollaire 4.1. Toutes les représentations condensées basées sur le support issues du cadre k-libre comme la représentation par non-dérivabilité, 0-libre ou disjonctif-libres [CG02b, CG03, BBR03] sont *inintéressantes* dans le cadre des motifs séquentiels.

Ce corollaire est très important dans le cadre des représentations condensées possibles pour les motifs séquentiels. Ainsi, seules les séquences ayant une fréquence de 0 (donc non-fréquentes) seront dérivables ou potentiellement 0-libres ou disjointe-libres, cette caractéristique ne baissera donc en rien la taille des séquences fréquentes extraites et ne sera donc d'aucune utilité dans le cas d'une représentation condensée.

6.3 Bornes supérieures pour la fréquence d'une séquence

Nous avons vu dans la sous-section précédente, qu'il était impossible de trouver une meilleure borne inférieure que 0 dans le cadre des motifs séquentiels. Qu'en est-il pour les bornes supérieures ? Est-ce que les meilleures bornes sont celles déjà utilisées par les algorithmes de type *Apriori* grâce aux règles se basant sur l'antimonotonie de la fréquence pour les supports ?

Nous allons montrer dans l'exemple suivant qu'il est possible de faire mieux que ces simples règles d'antimonotonie et que donc, **tous les algorithmes actuels d'extraction de motifs séquentiels ne sont pas optimaux en matière d'élagages de séquences**.

Exemple 4.16. Considérons l'ensemble de contraintes $\mathcal{CS}$ suivant :

$$\mathcal{CS} = \begin{cases} f_{\emptyset} = 1 \\ f_{\langle a \rangle} = 1 \\ f_{\langle b \rangle} = 1 \\ f_{\langle c \rangle} = 1 \\ f_{\langle (a)(b) \rangle} = \frac{1}{2} \\ f_{\langle (a)(c) \rangle} = 1 \\ f_{\langle (b)(c) \rangle} = \frac{1}{2} \end{cases}$$

Cet ensemble de contraintes est satisfiable. La base $\mathcal{D}$ suivante est un exemple d'une des bases possibles satisfaisant l'ensemble $\mathcal{CS}$:

$\mathcal{D} =$

Id. Séquences	Séquences
C_1	$(a)(c)(b)$
C_2	$(b)(a)(c)$

Grâce au *principe d'antimonotonie*, nous savons que $f_{\langle (a)(b)(c) \rangle} \leqslant f_{\langle (a)(b) \rangle}$, $f_{\langle (a)(b)(c) \rangle} \leqslant f_{\langle (a)(c) \rangle}$ et que $f_{\langle (a)(b)(c) \rangle} \leqslant f_{\langle (b)(c) \rangle}$. Donc dans ce cas là, $f_{\langle (a)(b)(c) \rangle} \leqslant \frac{1}{2}$.

Le *principe d'antimonotonie* est utilisé dans les algorithmes de types *Apriori* afin d'optimiser l'opération de génération, de comptage des candidats et permet d'élaguer (i.e. supprimer) les séquences non-fréquentes. Supposons maintenant que nous avons lancé un algorithme de ce type la afin d'extraire les séquences fréquentes de la base de données transactionnelles $\mathcal{D}$ avec une fréquence minimale de $\frac{1}{2}$.

Après la deuxième étape de comptage (pour les séquences de taille deux : $\langle (a)(b) \rangle$, $\langle (a)(c) \rangle$ et $\langle (b)(c) \rangle$), l'algorithme va passer à une étape de génération et essayer de générer, entres autres, la séquence $\langle (a)(b)(c) \rangle$ puis compter sa fréquence. Pourtant, cette étape de comptage pourraient être évitée puisqu'à partir de l'ensemble de contraintes de fréquences (qui représente les informations connues sur les séquences après la deuxième étape de comptage de l'algorithme) nous pouvons directement conclure que $f_{\langle (a)(b)(c) \rangle} = 0$. Pour comprendre cela, nous allons illustrer le raisonnement

logique en plusieurs points :

1. Chaque séquence d'une transaction de $\mathcal{D}$ contenant les séquences $\langle(a)(c)\rangle$ et $\langle(b)\rangle$, contient aussi la séquence $\langle(a)(b)\rangle$ ou bien $\langle(b)(c)\rangle$ (ou même les deux).

2. Supposons maintenant que $f_{\langle(a)(b)(c)\rangle} = a > 0$ avec a un nombre rationnel, alors de par les conditions de $\mathcal{CS}$, il doit au moins y avoir une transaction T dans $\mathcal{D}$ telle que sa séquence ne supporte par la séquence $\langle(a)(b)\rangle$ ou $\langle(b)(c)\rangle$ (puisque les fréquences de ces séquences sont de $\frac{1}{2}$.

3. Or, comme $f_{\langle b\rangle} = 1$ et $f_{\langle(a)(c)\rangle} = 1$, la séquence de la transaction T doit aussi contenir $\langle(a)(c)\rangle$ et $\langle(b)\rangle$, et donc, à cause du premier point, elle doit aussi contenir $\langle(a)(b)\rangle$ ou $\langle(b)(c)\rangle$. D'où contradiction avec le deuxième point. Donc $f_{\langle(a)(b)(c)\rangle}$ doit être égal à 0.

L'exemple précédent illustre bien l'idée que l'on peut obtenir des règles plus précises que celles de l'antimonotonie. De plus, cet exemple permet aussi d'exhiber, encore une fois, le lien entre les classes d'équivalences et la dérivation de règles puisque, traduit autrement, dans notre exemple, nous essayons de trouver une séquence s qui supporte uniquement les séquences $\langle(a)(c)\rangle$ et $\langle b\rangle$ (et leurs sous-séquences). Or, il est impossible de construire cette séquence s puisque $\mathcal{E}(\mathcal{S}', \mathcal{S}) = \emptyset$ avec :

$$\mathcal{S} = \left\{ \begin{array}{c} \langle(a)(b)(c)\rangle \\ \langle(a)(b)\rangle \\ \langle(a)(c)\rangle \\ \langle(b)(c)\rangle \\ \langle(a)\rangle \\ \langle(b)\rangle \\ \langle(c)\rangle \\ \langle()\rangle \end{array} \right\} \text{ et } \mathcal{S}' = \left\{ \begin{array}{c} \langle(a)(c)\rangle \\ \langle(b)\rangle \\ \langle(c)\rangle \\ \langle(a)\rangle \\ \langle()\rangle \end{array} \right\}$$

Afin d'automatiser le processus d'extraction de règles de dérivations plus strictes que celles se basant sur l'antimonotonie, nous proposons l'utilisation d'une procédure dite *d'élimination* : la méthode d'élimination de Fourier-Motzkin.

En effet dans le cadre de l'extraction d'itemsets, le système d'équations linéaires associé à un ensemble de contraintes de fréquences pour les itemsets est simple à décrire (voir exemple 4.9) puisque le nombre d'équations est égal au nombre de variables. Un moyen efficace de résoudre ce

système est donné par l'élimination de Gauss-Jordan [Atk89] ou par la décomposition de Cholesky [GVL96].

Dans le cadre des motifs séquentiels le système d'équations linéaires associé à un ensemble de contraintes de fréquences ne peut pas être résolu à travers ces méthodes d'éliminations classiques puisque le système possède plus de variables que d'équations. Nous devons donc éliminer certaines variables du système.

La méthode d'élimination de Fourier-Motzkin

Une méthode d'élimination (généralement appelé ∃-élimination) d'un ensemble de variables V d'un système d'inégalités linéaires consiste à créer un nouveau système d'inégalités linéaires qui ne contient pas les variables de l'ensemble V tel que ce système possède les mêmes solutions sur l'ensemble des variables restantes.

La méthode d'élimination de Fourier-Motzkin fonctionne de la manière suivante :

Considérons un système $Sys(F)$ contenant n inégalités et r variables $x_1 \dots x_r$ avec $r > n$.

$$Sys(F) = \begin{cases} a_{11}x_1 + \ldots + a_{r1}x_r \geqslant b_1 \\ \vdots \\ a_{1n}x_1 + \ldots + a_{rn}x_r \geqslant b_1 \end{cases}$$

Supposons que l'on veuille éliminer la variable x_r (i.e. $V = \{x_r\}$). Les inégalités du système $Sys(F)$ peuvent alors être groupées en trois ensembles, selon le signe du coefficient de la variable x_r (positif, négatif, nul) :

1. L'ensemble P qui contient toutes les inégalités telles que $x_r \geqslant \sum_{k=1}^{r-1} a_k x_k$
2. L'ensemble N qui contient toutes les inégalités telles que $x_r \leqslant \sum_{k=1}^{r-1} a_k x_k$
3. L'ensemble Z contenant les inégalités où x_r ne joue aucun rôle.

L'élimination consiste à produire un système $\exists_{x_r}$-$Sys(F)$ équivalent à $Sys(F)$, c'est-à-dire que le nouveau système possède une solution si et seulement si $Sys(F)$ possède aussi une solution.

Le nouveau système $\exists_{x_r}$-$Sys(F)$ contient ainsi toutes les inégalités de l'ensemble Z (les inégalités ou x_r ne joue aucun rôle) en plus de toutes les combinaisons possibles $p \leqslant n$ entre les deux premières classes d'inégalités, avec $n \in N$ et $p \in P$. Cette méthode a dans le pire des cas une complexité exponentielle. Notons que le système $\exists_{x_r}$-$Sys(F)$ contient ainsi $(n - n_P - n_N) + n_N.n_P$

inégalités, avec $n_P = |P|$ et $n_N = |N|$. Dans le cas où $n_P = n_N = \frac{n}{2}$, le nombre d'inégalités dans $\exists_{x_r}$-$Sys(F)$ est de $\frac{n^2}{4}$.

Le principal problème de cette méthode est la redondance des inégalités. Par exemple, l'inégalité $x_1 + x_2 \geqslant x_3 + x_4$ est redondante par rapport aux inégalités $x_1 \geqslant x_3$ et $x_2 \geqslant x_4$. Ainsi, il n'est pas rare de voir dans les cas pratiques un taux de redondance noyant les inégalités intéressantes.

Une interprétation graphique du problème de redondance est donnée Figure 4.5.

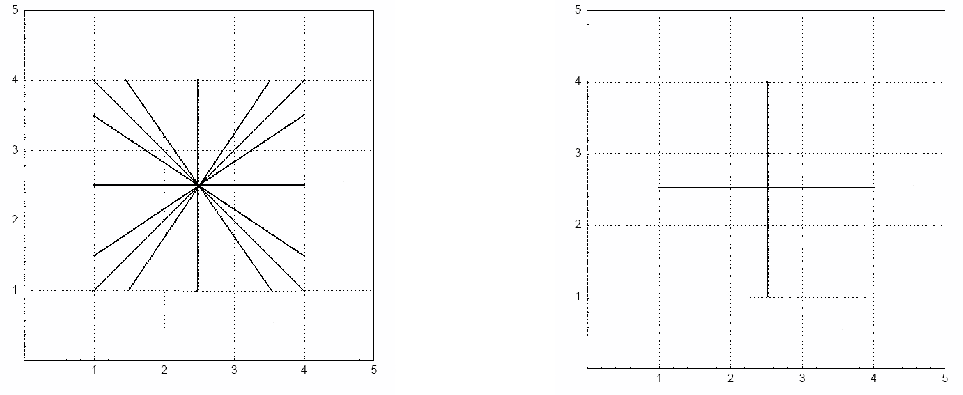


FIG. 4.5: Un système d'inégalités redondantes et le même système simplifié

L'exemple suivant illustre la méthode d'élimination :

Exemple 4.17. Considérons le système d'inégalités suivant :

$$Sys(F) = \begin{cases} 2x_1 - 11x_2 & \leqslant 3 \\ -3x_1 + 2x_2 & \leqslant -5 \\ x_1 + 3x_2 & \leqslant 4 \\ -2x_1 & \leqslant -3 \end{cases}$$

Nous allons éliminer la variable x_2 du système. Isolons d'abord la variable x_2 :

1. $P = \{11x_2 \geqslant 2x_1 - 3\}$
2. $N = \{2x_2 \leqslant 3x_1 - 5 \ ; \ 3x_2 \leqslant -x_1 + 4\}$
3. $Z = \{-2x_1 \leqslant -3\}$

Le nouveau système est alors :

$$\exists_{x_r}\text{-}Sys(F) = \begin{cases} 4x_1 - 6 & \leqslant 33x_1 - 55 \\ 6x_1 - 9 & \leqslant -11x_1 + 44 \\ & \\ -2x_1 & \leqslant -3 \end{cases}$$

D'où finalement :

$$\exists_{x_r}\text{-}Sys(F) = \left\{ \begin{array}{rl} x_1 & \geqslant \frac{49}{29} \\ x_1 & \leqslant \frac{53}{17} \\ -2x_1 & \leqslant -3 \end{array} \right.$$

Le système $\exists_{x_r}\text{-}Sys(F)$ est équivalent au système $Sys(F)$ et il ne contient plus la variable x_2.

Considérons maintenant l'emploi de cette méthode d'élimination dans le cadre de la découverte de règles donnant une borne supérieures pour la fréquence d'une séquence. Supposons que l'on veuille extraire les règles de dérivations possibles pour la séquence $\langle(a)(b)\rangle$. L'exemple suivant illustre les différentes étapes d'éliminations successives afin d'extraire les règles possibles.

Exemple 4.18 (Règles de dérivations pour les séquences $\langle(a)(b)\rangle$ et $\langle(a)(b)(c)\rangle$)**.**

Soit $\mathcal{S}$ l'ensemble des séquences présentes dans l'ensemble de contraintes de fréquences $\mathcal{CS}$:

$$\mathcal{S} = \left\{ \begin{array}{c} \langle(a)(b)\rangle \\ \langle(b)\rangle \\ \langle(a)\rangle \\ \langle()\rangle \end{array} \right\}$$

Les sous-ensembles $\mathcal{S}'$ de $\mathcal{S}$ tels que $\mathcal{E}(\mathcal{S}', \mathcal{S}) \neq \emptyset$ sont :

$$\begin{array}{rcl} \mathcal{S}_1 & = & \{\langle()\rangle\} \\ \mathcal{S}_2 & = & \{\langle()\rangle, \langle(a)\rangle\} \\ \mathcal{S}_3 & = & \{\langle()\rangle, \langle(b)\rangle\} \\ \mathcal{S}_4 & = & \{\langle()\rangle, \langle(a)\rangle, \langle(b)\rangle\} \\ \mathcal{S}_5 & = & \{\langle()\rangle, \langle(a)\rangle, \langle(b)\rangle, \langle(a)(b)\rangle\} \end{array}$$

L'existence d'une base de données satisfaisant $\mathcal{CS}$ est équivalent à l'existence d'une solution positive du système d'équations linéaires suivant :

$$Sys(\mathcal{CS}) = \left\{ \begin{array}{rcll} 1 & = & X_1 + X_2 + X_3 + X_4 + X_5 & \\ f_{\langle(a)\rangle} & = & X_2 + X_4 + X_5 & \\ f_{\langle(b)\rangle} & = & X_3 + X_4 + X_5 & \\ f_{\langle(a)(b)\rangle} & = & X_5 & \\ X_i & \geqslant & 0 \qquad \forall i \in \{1, 2 \ldots, 5\} & \end{array} \right.$$

Ce système possède une solution positive si et seulement si le système équivalent $Sys(\mathcal{CS})_1$ possède lui-même une solution positive :

$$Sys(\mathcal{CS})_1 = \begin{cases} 1 & \geqslant & X_2 + X_3 + f_{\langle(a)(b)\rangle} + X_4 \\ f_{\langle(a)\rangle} & = & X_2 + f_{\langle(a)(b)\rangle} + X_4 \\ f_{\langle(b)\rangle} & = & X_3 + f_{\langle(a)(b)\rangle} + X_4 \\ X_i & \geqslant & 0 \qquad \forall i \in \{2,3,4\} \end{cases}$$

Commençons par éliminer la variable X_2, on sait que : $X_2 = f_{\langle(a)\rangle} - f_{\langle(a)(b)\rangle} - X_4$. On a alors le système d'équations équivalent suivant

$$Sys(\mathcal{CS})_2 = \begin{cases} 0 & \leqslant & f_{\langle(a)\rangle} - f_{\langle(a)(b)\rangle} - X_4 \\ 1 & \geqslant & f_{\langle(a)\rangle} + X_3 \\ f_{\langle(b)\rangle} & = & X_3 + f_{\langle(a)(b)\rangle} + X_4 \end{cases}$$

De la même manière, nous pouvons éliminer la variable X_3 par substitution :

$$Sys(\mathcal{CS})_3 = \begin{cases} 0 & \leqslant & f_{\langle(a)\rangle} - f_{\langle(a)(b)\rangle} - X_4 \\ 0 & \leqslant & f_{\langle(b)\rangle} - f_{\langle(a)(b)\rangle} - X_4 \\ 1 & \geqslant & f_{\langle(a)\rangle} + f_{\langle(b)\rangle} - f_{\langle(a)(b)\rangle} - X_4 \end{cases}$$

Afin d'éliminer la variable X_4 nous procédons à une élimination par la méthode de Fourier-Motzkin. Les différents ensembles d'inégalités après isolation de la variable X_4 sont :

1. $P = \{X_4 \geqslant f_{\langle(a)\rangle} + f_{\langle(a)\rangle} - f_{\langle(a)(b)\rangle} - 1\}$; $X_4 \geqslant 0$
2. $N = \{X_4 \leqslant f_{\langle(a)\rangle} - f_{\langle(a)(b)\rangle}$; $X_4 \leqslant f_{\langle(b)\rangle} - f_{\langle(a)(b)\rangle}\}$
3. $Z = \emptyset$

D'où :

$$\exists_{X_4}\text{-}Sys(\mathcal{CS})_3 = \begin{cases} f_{\langle(a)\rangle} + f_{\langle(b)\rangle} - f_{\langle(a)(b)\rangle} - 1 & \leqslant & f_{\langle(a)\rangle} - f_{\langle(a)(b)\rangle} \\ 0 & \leqslant & f_{\langle(a)\rangle} - f_{\langle(a)(b)\rangle} \\ f_{\langle(a)\rangle} + f_{\langle(b)\rangle} - f_{\langle(a)(b)\rangle} - 1 & \leqslant & f_{\langle(b)\rangle} - f_{\langle(a)(b)\rangle} \\ 0 & \leqslant & f_{\langle(b)\rangle} - f_{\langle(a)(b)\rangle} \\ f_{\langle(a)(b)\rangle} & \geqslant & 0 \end{cases}$$

Ce qui équivaut au système final :

$$\exists_{X_4}\text{-}Sys(\mathcal{CS})_{final} = \begin{cases} f_{\langle(b)\rangle} & \leqslant & 1 \\ f_{\langle(a)(b)\rangle} & \leqslant & f_{\langle(a)\rangle} \\ f_{\langle(a)\rangle} & \leqslant & 1 \\ f_{\langle(a)(b)\rangle} & \leqslant & f_{\langle(b)\rangle} \\ f_{\langle(a)(b)\rangle} & \geqslant & 0 \end{cases}$$

On voit donc bien, à travers cet exemple, que l'ensemble des règles de dérivations pour la séquence ne contient pas de règles non-triviales plus intéressantes que les règles d'antimonotonies puisque nous avons :

$$0 \leqslant f_{\langle(a)(b)\rangle} \leqslant f_{\langle(a)\rangle}, f_{\langle(b)\rangle} \leqslant 1$$

Dans le cas de la séquence $\langle(a)(b)(c)\rangle$, le système final est :

$$\exists\text{-}Sys(\mathcal{CS})_{final} = \begin{cases} f_{\langle(a)(b)(c)\rangle} & \geqslant & 0 \\ f_{\langle(b)(c)\rangle} & \geqslant & f_{\langle(a)(b)(c)\rangle} \\ f_{\langle(a)(b)\rangle} & \geqslant & f_{\langle(a)(b)(c)\rangle} \\ f_{\langle(a)(c)\rangle} & \geqslant & f_{\langle(a)(b)(c)\rangle} \\ f_{\langle(c)\rangle} & \geqslant & f_{\langle(b)(c)\rangle} \\ f_{\langle(b)\rangle} & \geqslant & f_{\langle(b)(c)\rangle} \\ f_{\langle(a)\rangle} & \geqslant & f_{\langle(a)(c)\rangle} \\ f_{\langle(a)\rangle} & \geqslant & f_{\langle(a)(b)\rangle} \\ f_{\langle(c)\rangle} & \geqslant & f_{\langle(a)(c)\rangle} \\ f_{\langle(b)\rangle} & \geqslant & f_{\langle(a)(b)\rangle} \\ f_{\langle()\rangle} & \geqslant & f_{\langle(c)\rangle} \\ f_{\langle()\rangle} & \geqslant & f_{\langle(b)\rangle} \\ f_{\langle()\rangle} & \geqslant & f_{\langle(a)\rangle} \\ f_{\langle()\rangle} + f_{\langle(a)(b)\rangle} + f_{\langle(b)(c)\rangle} & \geqslant & f_{\langle(a)(b)(c)\rangle} + f_{\langle(a)(c)\rangle} + f_{\langle(b)\rangle} \end{cases}$$

On remarque alors que l'inégalité $f_{\langle()\rangle} + f_{\langle(a)(b)\rangle} + f_{\langle(b)(c)\rangle} \geqslant f_{\langle(a)(b)(c)\rangle} + f_{\langle(a)(c)\rangle} + f_{\langle(b)\rangle}$ est bien une règle de dérivation de support non-triviale (non-redondante) et permet de borner la fréquence de la séquence $\langle(a)(b)(c)\rangle$:

$$f_{\langle()\rangle} - f_{\langle(b)\rangle} + f_{\langle(a)(b)\rangle} + f_{\langle(b)(c)\rangle} - f_{\langle(a)(c)\rangle} \geqslant f_{\langle(a)(b)(c)\rangle}$$

Pour s'en convaincre comparons le résultat de cette règle par rapport aux règles d'antimonotonie dans la base de données transactionnelles suivante :

$\mathcal{D} =$

Id. Séquences	Séquences
C_1	$(a)(c)(b)$
C_2	$(b)(a)(c)$

Ainsi, $f_{\langle(a)(b)(c)\rangle} \leqslant f_{(a)(b)}$, $f_{\langle(a)(b)(c)\rangle} \leqslant f_{(a)(c)}$ et $f_{\langle(a)(b)(c)\rangle} \leqslant f_{(b)(c)}$, ce qui implique que $f_{\langle(a)(b)(c)\rangle} \leqslant \frac{1}{2}$. Pourtant avec $f_{\langle()\rangle} - f_{\langle(b)\rangle} + f_{\langle(a)(b)\rangle} + f_{\langle(b)(c)\rangle} - f_{\langle(a)(c)\rangle} \geqslant f_{\langle(a)(b)(c)\rangle}$, on a une meilleure borne puisque dans ce cas là : $f_{\langle(a)(b)(c)\rangle} \leqslant 0$.

L'algorithme d'énumération des classes d'équivalences non-vides ainsi que la méthode d'élimination de Fourier-Motzkin ont été implémentées conjointement avec T. Calders dans le langage Python et permettent actuellement d'extraire des règles non-triviales pour les séquences. Notre implémentation nous permet d'extraire des règles pour des séquences de taille maximum 5, au delà, le nombre de redondances et le temps de calcul nécessaire à la résolution du système d'équation linéaire associé rendent à la tâche presque impossible. Un ensemble de règles pour certaines séquences est présenté dans l'Appendix A.

7 Discussion

Motif	B. inférieures	B. Supérieures	Représentations condensées
Itemsets	variables, formule close	variables, formule close	NDI, k-libres
Séquences	0	variables, par éliminations	aucune

Tab. 4.1: Synthèse de la méthode de dérivation de règles pour les itemsets et les séquences

Dans ce chapitre, nous avons abordé la problématique des représentations condensées pour les motifs séquentiels. Nous avons jeté les bases formelles pour la compréhension du problème de satisfiabilité de contraintes de fréquences `SEQFREQSAT` en introduisant une relation d'équivalence sur le support d'une séquence ainsi que des classes d'équivalences basées sur cette même relation. Ces classes permettent de regrouper les séquences selon les sous-séquences qu'elles supportent.

Nous montrons ainsi dans ce chapitre que le problème de décider si une classe d'équivalence est non-vide ainsi que le problème `SEQFREQSAT` sont **NP**-complets et montrons les liens entre ces problèmes de décisions et le domaine de l'algèbre linéaire.

Le problème `SEQFREQSAT` nous a ainsi permis d'étudier dans la deuxième partie de ce chapitre, la possibilité de construire de nouvelles représentations condensées pour les motifs séquentiels. Les résultats de cette étude ainsi qu'une comparaison dans le cadre des itemsets sont donnés dans le Tableau 4.1. On remarquera que le calcul des bornes dans le cadre des itemsets est très

efficace, puisqu'il se base sur une formule close (principe d'inclusion-exclusion), or, dans le cadre des séquences, et comme nous l'avons vu tout au long de ce chapitre, les bornes sont calculées par rapport aux classes d'équivalences vides ou non-vides, cette caractéristique rend la possibilité de trouver une formule close très difficile pour les règles de séquences. Nous reviendrons sur ce point dans les perspectives associées à ce travail.

L'impossibilité de présenter une borne inférieure sur la fréquence meilleure que 0 nous permet d'affirmer que toutes les représentations condensées basées sur le support issues du cadre k-libre comme la représentation par non-dérivabilité, 0-libre ou disjonctif-libres [CG02b, CG03, BBR03] sont *inintéressantes* dans le cadre des motifs séquentiels. Ainsi, seules les séquences ayant une fréquence de 0 seront dérivables ou potentiellement 0-libres ou disjointe-libres rendant cette caractéristique inutile le cas d'une représentation condensée. Le cas des séquences 1-libres (dans le sens δ-libre) reste encore ouvert.

Chapitre 5

Motifs Conjonctifs Séquentiels

L'extraction de motifs séquentiels permet d'extraire des connaissances basées généralement sur un principe de temporalité. Ce type de connaissances séquentielles est souvent utilisé par des méthodes d'apprentissages supervisées et permet ainsi des applications dans le domaine de la *prédiction*. Dans ce chapitre, nous nous intéressons à la notion de *corrélations statistiques entre séquences*. Ce type de corrélations est très peu étudié dans le cadre des séquences contrairement au domaine des itemsets et des règles d'associations. Ainsi, dans ce chapitre, nous proposons un nouveau type de motifs permettant la transposition de ces corrélations dans le cadre de motifs séquentiels.

Une partie des travaux présentés dans ce chapitre a été publiée dans la conférence *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD 2008)* et le journal *Data Mining and Knowledge Discovery Journal*.

1 Introduction

Le problème d'extraction des règles d'associations a été introduit dans [AIS93]. Le but de cette méthode est d'obtenir toutes les relations causales entre les attributs d'une base de données en se basant sur des métriques statistiques tel que la confiance, la conviction ou le lift. Alors qu'il s'agissait d'un des objectifs des règles d'association, si nous considérons les motifs séquentiels, nous pouvons constater qu'il existe peu de travaux qui se sont intéressés à de telles règles. Aussi dans ce chapitre, nous souhaitons répondre à la question suivante : *Comment peut-on extraire des relations causales sur ces motifs ?*

Comme nous le disions précédemment, il existe à notre connaissance très peu de travaux qui se sont focalisés sur l'extraction de règles pour les motifs séquentiels [BCG07]. Pourtant comme cela est décrit dans [LZO99] et [ZXY08], les séquences pourraient être utilisées par les algorithmes

de classifications supervisées afin de développer des applications dans le domaine de la *prédiction*. Comme par exemple, la prédiction médicale [PK03], la prédiction de mots ou de phrases [NJ07] (dans les correcteurs orthographique et les éditeurs de textes) ou la prédiction de catastrophe (*"Un tremblement de terre de magnitude 6, suivi d'une secousse dans dans des régions côtières est généralement suivi à 23% par un raz-de-marée dans les 4 heures qui suivent la première alerte"*). Le faible intérêt des chercheurs pour cette problématique s'explique facilement par la difficulté de formalisation nécessaire dans le cadre des motifs semi-structurés ou structurés, ainsi, *comment peut-on transposer les notions ensemblistes utilisées dans le cadre des itemsets et des règles d'associations à des motifs possédant une structure ?* Par exemple, pour une séquence $\langle(a)(b,c)(d)\rangle$ et ses différentes sous-séquences, une relation triviale, mais possible, serait de dire que $\langle(a)(b,c)\rangle \Rightarrow \langle(d)\rangle$ mais qu'en est-il alors de $\langle(b,c)\rangle \Rightarrow \langle(a)(d)\rangle$? Cette deuxième relation causale n'est pas identique à la première, elle ne se base pas dans son implication sur la relation de temporalité, mais elle contient aussi un certain degré de connaissance. *Comment alors extraire de telles règles sans tomber dans le piège de l'explosion combinatoire ?*

Nos contributions dans ce chapitre sont doubles :

1. Tout d'abord nous introduisons un nouveau type de motifs à extraire sur des bases de données transactionnelles et discutons les différents aspects de la nouvelle tâche d'extraction.
2. Nous étudions la possibilité d'extraire des motifs non-redondants (de la même manière que dans le cadre des itemsets) afin d'extraire le plus petit ensemble de règles d'associations séquentielles non-redondantes. Cette approche est basée sur un outil de la branche de la combinatoire : le théorème d'inversion de Moebius [IR90] et nous permet d'introduire un concept de non-dérivabilité sur les motifs conjonctifs séquentiels.

Le chapitre est organisé de la manière suivante. La section 2 présente les différentes définitions associées aux motifs conjonctifs séquentiels. La section 3 discute de la possibilité de construire une représentation condensée pour les motifs conjonctifs séquentiels en se basant sur la dérivation de règles pour la fréquences de ces motifs. Notre algorithme d'extraction de motifs conjonctifs séquentiels est présenté dans la section 4. Les expériences réalisées sont décrites et discutées dans la section 5. Nous concluons ce chapitre par une discussion.

2 Motifs conjonctifs séquentiels

Dans le chapitre précédent nous avons introduit une classe d'équivalence pour les motifs séquentiels basée sur une relation d'équivalence définie sur la notion de support. Le but de ce chapitre est de presenter un motif ayant des liens avec ces classes d'équivalences mais ayant surtout des caractéristique permettant des applications pratiques.

2.1 Définitions

Définition 5.1 (Motif conjonctif séquentiel)**.** Un motif conjonctif séquentiel (MCS) est un sous-ensemble C de l'ensemble de toutes les séquences possibles $\mathcal{T}$ tel que pour tout $S \neq S' \in C$, S et S' sont incomparables (i.e. ni $S \not\preceq S'$ ni $S' \not\preceq S$). L'ensemble de tous les MCS est noté $\mathcal{C}$.

Exemple 5.1. Soit $s = \langle (a)(b,c) \rangle$ et $s' = \langle (b)(c) \rangle$ des séquences telles que $s \not\preceq s$ et $s' \preceq s$, alors C est un motif conjonctif séquentiel contenant ces deux séquence :

$$C = \{\langle (a)(b,c) \rangle, \langle (b)(c) \rangle\}$$

Définition 5.2 (Support d'un motif conjonctif séquentiel)**.** Une séquence $T \in \mathcal{T}$ supporte un MCS si pour chaque séquence $S \in C$, $S \preceq T$.

Le *support* d'un MCS C dans une base de données transactionnelles, noté $Support_{\mathcal{D}}(C)$, est le nombre de séquences des transactions de $\mathcal{D}$ qui satisfont (supportent) C.

Soient $C_1, C_2 \in \mathcal{C}$. C_1 est un sous-motif de C_2, noté $C_1 \preceq C_2$, si et seulement si pour toute séquence $S_1 \in C_1$, il existe une séquence $S_2 \in C_2$ tel que $S_1 \preceq S_2$.

Définition 5.3 (Règle d'association séquentielle)**.** Une règle d'association séquentielle est une implication de la forme : $S \Rightarrow S'$, avec S, S' des MCS et $S \cap S' = \emptyset$.

La confiance d'une règle est définie telle que :

$$Conf(S \Rightarrow S') = \frac{Support_{\mathcal{D}}(S \cup S')}{Support_{\mathcal{D}}(S)}$$

Exemple 5.2. Soient $C_1 = \{\langle (a)(b) \rangle, \langle (a,c) \rangle\}$ et $C_2 = \{\langle (b) \rangle, \langle (c) \rangle\}$ des motifs conjonctifs séquentiels et soit $\mathcal{D}$ la base de données transactionnelles suivante :

$\mathcal{D} =$

Id. Séq	Séquences
S_1	$(a)(b)(a,c)$
S_2	$(a,b)(c)$
S_3	$(c)(a)$

- C_2 est un sous-motif de C_1 (i.e. $C_2 \preceq C_1$) puisque $\langle (b) \rangle \preceq \langle (a)(b) \rangle$ et $\langle (c) \rangle \preceq \langle (a,c) \rangle$
- $Support_{\mathcal{D}}(C_1) = 1$ car seule la séquence $S_1 = \langle (a)(b)(a,c) \rangle$ supporte C_1.
- $Support_{\mathcal{D}}(C_2) = 2$ puisque les séquences S_1 et S_2 supportent C_2.

A partir des définitions précédentes, le lemme suivant est immédiat.

Lemme 5.1 (Antimonotonie)**.** Soient C_1, C_2 des MCS. Si $C_1 \preceq C_2$, alors, pour toute base de données transactionnelles $\mathcal{D}$:

$$Support(C_1) \geqslant Support(C_2)$$

Le lemme 5.1 est très important puisque il permet à notre algorithme d'exploiter la propriété d'antimonotonie afin d'élaguer plus efficacement l'espace de recherche. La question de la génération optimale de candidats est discutée dans le paragraphe et les définitions suivantes.

Soit $P \subseteq \mathcal{T}$ un ensemble de séquences. $\lceil P \rceil$ représente le MCS tel que $\{S \in P \mid \nexists S' \in P : S' \prec S\}$. $\downarrow P$ représente la fermeture transitive de P (i.e. l'ensemble $\{S \in \mathcal{T} \mid \exists S' \in P : S' \preceq S\}$).

Soit C_1 et C_2 des MCS. C_2 *couvre* C_1, noté $C_1 \rightarrow C_2$, si $C_1 \prec C_2$ et qu'ils n'existent pas d'autres MCS C_3 tels que $C_1 \prec C_3 \prec C_2$. On dit alors que C_2 est une *spécialisation directe* de C_1, de même, C_1 est une généralisation directe de C_2. Dans le cadre de l'extraction de motifs séquentiels, la génération d'un ensemble de séquences spécialisées pour une séquence s est triviale. En effet, étant donnée une séquence $s = \langle it_1, \ldots, it_n \rangle$ avec n itemsets, les généralisations directes de s, notées $dg(s)$, sont l'ensemble de séquences suivant :

$$\bigcup_{\substack{j=1 \\ |it_j|>1}}^{n} \{\langle it_1, \ldots, it_{j-1}, it_j \setminus \{i\}, it_{j+1}, \ldots, it_n \rangle \mid i \in it_j\} \cup \bigcup_{\substack{j=1 \\ |it_j|=1}}^{n} \{\langle it_1, \ldots, it_{j-1}, it_{j+1}, \ldots, it_n \rangle \mid i \in it_j\} \ .$$

Dans le cadre des motifs conjonctifs séquentiels, et comme le montre le lemme suivant, la généralisation (et la spécialisation) n'est pas aussi trivial.

Lemme 5.2 (Généralisation et Spécialisation)**.** Soit C un MCS. L'ensemble de généralisations directes de C est l'ensemble de MCS suivant :

$$\{\lceil (C \setminus S) \cup dg(S) \rceil \mid S \in C\}$$

L'ensemble de spécialisations directes de C est l'ensemble de MCS suivant :

$$\{C \cup \{S\} \mid S \notin C, dg(S) \subseteq C\}$$

Démonstration. La preuve se base sur le fait que C_1 est une généralisation de C_2 si et seulement si $\downarrow C_1 \subseteq \downarrow C_2$. Si de plus, $\downarrow C_1 = \downarrow C_2$, alors ceci implique que $C_1 = C_2$. En fait, pour l'ensemble de généralisations directes du MCS C, il est facile de voir que pour chaque séquence $S \in C$, $\downarrow (C \setminus S) \cup dg(S) \subseteq \downarrow C$ et que $\downarrow C \setminus \downarrow (C \setminus S) \cup dg(S) = \{S\}$.

□

Ce lemme permet ainsi de générer l'ensemble de motifs candidats en partant des motifs les plus généraux aux plus spécialisés. Cette génération par étape permet ainsi d'exploiter la propriété d'antimonotonie du support et de développer des méthodes d'extractions basés sur l'algorithme générique d'extraction de motifs (i.e. en se basant sur l'approche *générer-élaguer*).

L'exemple suivant illustre une extraction par niveaux de motifs conjonctifs séquentiels.

Exemple 5.3. Soit $\mathcal{D}$ la base de données transactionnelles suivante :

$\mathcal{D} =$

Id. Séq	Séquences
S_1	$(a,b)(b)$
S_2	$(a)(b)(c)$

Supposons que le support minimal $\sigma = 1$. Alors à la première étape de l'extraction nous allons avoir comme MCS fréquents :

1. Étape 1 : $\{\langle(a)\rangle\} : 2$, $\{\langle(b)\rangle\} : 2$ et $\{\langle(c)\rangle\} : 1$
2. Étape 2 : $\{\langle(a)\rangle, \langle(b)\rangle\} : 2$
3. Étape 3 : $\{\langle(a,b)\rangle\} : 1$, $\{\langle(a)(b)\rangle\} : 1$, $\{\langle(b)(a)\rangle\} : 1$ et $\{\langle(b)(b)\rangle\} : 1$
4. Étape 4 : $\{\langle(a,b)\rangle, \langle(b)(b)\rangle\} : 1$, $\{\langle(a)\rangle, \langle(b)(b)\rangle\} : 1$ et $\{\langle(a)(b)\rangle, \langle(b)(b)\rangle\} : 1$

Les règles de confiance minimum $\frac{1}{2}$ issues de ces MCS sont :

1. $\langle(a)\rangle \Rightarrow \langle(b)\rangle$ $(Confiance = 1)$
2. $\langle(b)\rangle \Rightarrow \langle(a)\rangle$ $(Confiance = 1)$
3. $\langle(a)(b)\rangle \Rightarrow \langle(b)(b)\rangle$ $(Confiance = 1)$
4. $\langle(b)(b)\rangle \Rightarrow \langle(a)(b)\rangle$ $(Confiance = 1)$
5. $\langle(a,b)\rangle \Rightarrow \langle(b)(b)\rangle$ $(Confiance = 1)$
6. $\langle(b)(b)\rangle \Rightarrow \langle(a,b)\rangle$ $(Confiance = 1)$
7. $\langle(b)(b)\rangle \Rightarrow \langle(a)\rangle$ $(Confiance = 1)$
8. $\langle(a)\rangle \Rightarrow \langle(b)(b)\rangle$ $(Confiance = \frac{1}{2})$

3 Motifs Conjonctifs Séquentiels Non-dérivables

Nous avons vu dans les deux chapitres précédents que les représentations condensées permettaient d'avoir un ensemble de motifs fréquents réduit tout en gardant la même pouvoir d'expression.

Dans cette section, nous montrons qu'il est possible d'extraire un ensemble de motifs conjonctifs séquentiels non-redondants en se basant sur un concept de non-dérivabilité déjà utilisé dans le cadre d'extraction des itemsets. Bien que cette représentation soit impossible dans le cadre des séquences (voir le chapitre précédent), elle est tout à fait valable dans le cadre des motifs conjonctifs séquentiels.

Le théorème suivant montre qu'un ensemble de MCS muni de la relation d'ordre partielle $\preceq$ forme un treillis. Ce théorème souligne le contraste avec l'ensemble des séquences qui ne peut être exprimé qu'au travers d'un hyper-treillis. Ainsi par exemple, les séquences $\langle (a,b)(a)\rangle$ et $\langle (a)(a,b)\rangle$ n'ont pas une unique borne supérieure mais bien deux : $\{\langle (a),(a)\rangle$ et $\langle (a,b)\rangle\}$. Dans l'ensemble MCS, cette borne supérieure est unique : $\{\langle (a)(a)\rangle\,;\langle (a,b)\rangle\}$. La figure 5.1 illustre un exemple de treillis ayant pour élément maximal le MCS $\{\langle (a)(b)(a)\rangle\}$.

Théorème 5.1. Pour l'ensemble des MCS $\mathcal{C}$, $(\downarrow \mathcal{C}, \preceq)$ est un treillis.

Démonstration. Soient $C_1, C_2 \in \mathcal{C}$. Il est facile de voir que les deux ensemble suivants forment les *uniques* bornes supérieures et inférieures (*meet* et *joins*) de C_1 et C_2 :

$$\bigwedge \lceil \downarrow C_1 \cap \downarrow C_2 \rceil \quad \text{et} \quad \bigwedge \lceil C_1 \cup C_2 \rceil$$

□

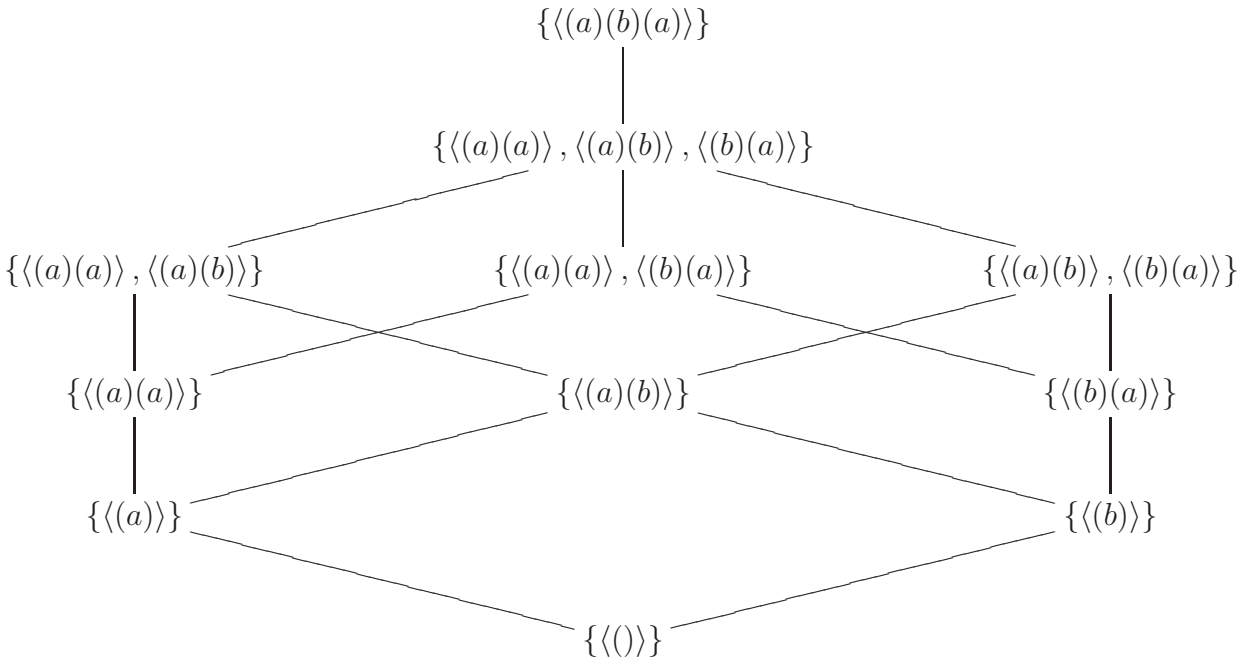


Fig. 5.1: Exemple d'un treillis ayant pour élément maximal le MCS $\{\langle (a)(b)(a)\rangle\}$

Le fait que $(\mathcal{C}, \preceq)$ soit un treillis ouvre un ensemble de possibilités et de méthodes mathématiques utiles dans le cadre de la construction de représentations condensées. Nous allons montrer maintenant comment utiliser le principe d'inversion de Möbius [IR90] afin de générer un ensemble de règles permettant de dériver le support d'un MCS à partir de ses sous-motifs d'une manière très proche que celle utilisée dans le cadre de l'extraction d'itemsets non-dérivables. Dans la sous-section suivante, nous introduisons de manière formelle le principe d'inversion de Möbius.

3.1 Le principe d'inversion de Möbius

Le principe d'inversion de Möbius occupe une place prépondérante dans la branche des combinatoires en mathématiques. Bien que ce principe fut à la base énoncé par A. F. Möbius dans le cadre de la théorie des nombres, il est largement utilisé dans plusieurs autres branches telles que la topologie ou les probabilités.

Soit P un ensemble partiellement ordonné. Tout au long de ce paragraphe nous allons considérer des matrices α dont les lignes et les colonnes sont indexées par les différents éléments de l'ensemble P.

Définition 5.4 (Algèbre d'incidence [IR90])**.** L'algèbre d'incidence $\mathcal{U}(P)$ est formée des fonctions à valeurs (ou matrices) dans $\mathbb{R}$ $f(x, y)$ avec $x, y \in P$, ayant la propriété $f(x, y) = 0$ si $x \not\leqslant y$.

Par la définition de la multiplication des matrices nous avons :

$$(\alpha\beta)(x, y) = \sum_{z \in P} \alpha(x, z)\beta(z, y)$$

Si $\alpha, \beta \in \mathcal{U}(P)$, alors cette somme est finie car P est localement fini et z sera défini uniquement sur l'intervalle $[x, y]$ $(x \leqslant z \leqslant y)$. On vérifiera aisément que ce produit est associatif.

La fonction ζ qui jouera un rôle important dans notre représentation est définie telle que :

$$\zeta(x, y) = \begin{cases} 1 & \text{si } x \leqslant y \text{ dans P,} \\ 0 & \text{sinon.} \end{cases}$$

Proposition 5.1. La fonction ζ est inversible dans l'algèbre d'incidence $\mathcal{U}(P)$.

La fonction inverse, $\mu(x, y)$, appelée fonction de Möbius, est aisément calculable par récurrence : $\mu(x, y) = 1$, $\forall x \in P$, et si on connaît $\mu(x, z)$, $\forall z \in [x, y[$, $\mu\zeta = I$ (matrice identité) donne :

$$\mu(x, y) = -\sum_{x \leqslant z < y} \mu(x, z), \text{ pour } x < y \in P$$

Nous présentons l'exemple suivant afin d'illustrer ces définitions :

Exemple 5.4. Soit l'ensemble d'entiers $P = \{1, 2, 3, 4, 6, 12\}$, le treillis de tous les diviseurs positifs de l'entier 12. On a alors :

$$\zeta = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}, \ \mu = \begin{pmatrix} 1 & -1 & -1 & 0 & 1 & 0 \\ 0 & 1 & 0 & -1 & -1 & 1 \\ 0 & 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

La première ligne de μ a été calculée de la manière suivante :

- $\mu(1,1) = 1$
- $\mu(1,2) = -\mu(1,1) = -1$
- $\mu(1,3) = -\mu(1,1) = -1$
- $\mu(1,4) = -\mu(1,1) - \mu(1,2) = 0$
- $\mu(1,6) = -\mu(1,1) - \mu(1,2) - \mu(1,3) = 1$
- $\mu(1,12) = -\mu(1,1) - \mu(1,2) - \mu(1,3) - \mu(1,4) - \mu(1,6) = 0$

Théorème 5.2 (Formule d'inversion de Möbius [IR90])**.** Soit f et g des fonctions à valeurs dans $\mathbb{R}$ définie sur P telles que :

$$g(x) = \sum_{a \geqslant x} f(a)$$

pour tout $x \in P$. Alors pour tout $x \in P$:

$$f(x) = \sum_{a \geqslant x} \mu(x, a) g(a) \tag{5.1}$$

Nous allons maintenant voir comment ce principe peut être utilisé dans le cas d'une représentation condensée pour les motifs conjonctifs séquentiels.

3.2 Dérivation de règles pour le support

Soit C un MCS et $\mathcal{D}$ une base de données transactionnelles. Pour chaque sous-motif $C' \in \downarrow C$ dans le treillis, nous pouvons associer, comme pour les itemsets et les séquences, deux valeurs : tout d'abord, la valeur de support $s(C') = \mathit{Support}(C', \mathcal{D})$ et la valeur d'apparition $a(C') = A(C')$, avec

$$A(C') = |\{T \in \mathcal{D} \mid \forall C'' \in \downarrow C : T \models C'' \Leftrightarrow C'' \preceq C'\}|$$

Ainsi, comme pour les itemsets [Cal03] et les séquences (voir Chapitre 4), la valeur $a(C')$ représente le nombre de transactions dans $\mathcal{D}$ qui supportent **exactement** le MCS C'. De plus,

remarquons que chaque transaction comptée dans la valeur d'apparition $a(C')$ supporte un MCS uniquement si ce MCS est plus général que C'.

De ces définitions et remarque nous pouvons alors énoncer le lemme suivant qui exhibe une relation entre la valeur de support et la valeur d'apparition.

Lemme 5.3. Pour tout $C' \preceq C$,

$$s(C') \;=\; \sum_{C' \preceq C''} a(C'')$$

A partir de ce lemme soulignant la relation entre le support et l'apparition d'un MCS, et comme dans le cadre des itemsets et des motifs séquentiels, nous pouvons énoncer le théorème suivant sur la satisfiabilité d'un ensemble de contraintes sur le support de motifs conjonctifs séquentiels.

Théorème 5.3. Soit C un MCS. Associons pour tout sous-motif $C' \preceq C$, la valeur (entière) s_C.

Il existe au moins une base de données transactionnelles $\mathcal{D}$, telle que pour tout $C' \preceq C$, $\mathit{Support}(C', \mathcal{D}) = s_C$ si et seulement si le système d'équations suivant, ayant pour variables les apparitions $a(C')$ des sous-motifs $C' \preceq C$, possède une solution positive :

$$Sys(F) = \begin{cases} a(C') \;=\; 0 & \forall C' : \mathcal{E}(C', \downarrow C) = \emptyset \\ a(C') \;\geqslant\; 0 & \forall C' : \mathcal{E}(C', \downarrow C) \neq \emptyset \\ \sum_{C' \preceq C''} a(C'') \;=\; s'_C & \forall C' \preceq C \end{cases}$$

Le premier type d'équation dans ce système souligne bien le fait que si la classe d'équivalence $\mathcal{E}(C', \downarrow C)$ est vide, alors aucune transaction contenant une séquence supportant le MCS C' ne peut exister dans la base $\mathcal{D}$. Le deuxième type d'inégalités est une déduction immédiate de la définition d'une apparition d'un motif conjonctif séquentiel. Le troisième type d'inégalités représente le lien entre l'apparition et le support d'un MCS C' énoncé dans le Lemme 5.3.

De plus, pour chaque base de données transactionnelles satisfaisant pour chaque sous-motif $C' \preceq C$, $\mathit{Support}(C', \mathcal{D}) = s_C$, l'ensemble des apparitions $a(C') = a(C', \mathcal{D})$ est une solution pour le système.

Le principe d'inversion de Möbius énoncé dans le Théorème 5.2, nous permet d'affirmer que s'il existe une relation permettant d'exprimer le support d'un MCS C' en fonction des apparitions alors il existe une fonction μ permettant d'exprimer l'apparition d'un MCS C' en fonction des supports des super-motifs de C'. Cette affirmation est plus formellement énoncée dans le lemme suivant :

Lemme 5.4. Il existe une fonction μ qui permet d'associer à chaque paire de MCS $C', C'' \preceq C$ un entier $\mu(C', C'')$, tel que pour chaque base de données transactionnelles $\mathcal{D}$, nous avons :

$$a(C') \;=\; \sum_{C' \preceq C''} \mu(C', C'') s(C'')$$

De cette manière, la valeur d'apparition $a(C')$ peut être exprimée comme uniquement comme une combinaison linéaire des supports des super-motifs du MCS C'.

En combinant le Lemme 5.4 avec le Théorème 5.3 nous avons :

Théorème 5.4. Associons pour chaque MCS $C' \preceq C$, un entier s_C (le support). Il existe une base de données transactionnelles $\mathcal{D}$ satisfaisant pour chaque $C' \preceq C$, $\mathit{Support}(C', \mathcal{D}) = s_C$ si et seulement si :

$$\begin{cases} \sum_{C' \preceq C''} \mu(C', C'') s(C'') \;=\; 0 & \forall C' : \mathcal{E}(C', \downarrow C) = \emptyset \\ \sum_{C' \preceq C''} \mu(C', C'') s(C'') \;\geqslant\; 0 & \forall C' : \mathcal{E}(C', \downarrow C) \neq \emptyset \end{cases}$$

Ce théorème est important puisqu'il permet de dériver pour chacun des sous-motifs C' du MCS C une règle de dérivation comme dans le cas des itemsets non-dérivables. Nous présentons la dérivation de règles de support de manière pratique afin dériver le support d'un MCS dans l'exemple suivant.

Exemple 5.5. Soit une base de donnés transactionnelles :

$\mathcal{D} =$

Id. Séq	Séquences
S_1	$(a)(b)(c)$

Et supposons $\sigma = 1$ la valeur du support minimal.

Nous voulons dériver des règles sur le support du motif conjonctif suivant : $C = \{\langle (a) \rangle ; \langle (b) \rangle\}$ et supposons connus les supports des sous-motifs de C : $P = \{\{\langle () \rangle\}, \{\langle a \rangle\}, \{\langle b \rangle\}$. Le treillis $(\mathcal{L}, \preceq)$ construit sur l'ensemble $\{P \cup \{C\}\}$ peut être représenté sous la forme d'une matrice :

$$\zeta = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Cette matrice représente, outre la relation d'ordre, les différentes relations entre le support et les les apparitions des sur-motifs. Ainsi la première ligne qui traduit le fait que $\{\langle () \rangle\} \preceq \{\langle () \rangle\}$,

$\{\langle\rangle\} \preceq \{\langle(a)\rangle\}$, $\{\langle\rangle\} \preceq \{\langle(b)\rangle\}$ et $\{\langle\rangle\} \preceq \{\langle(a)\rangle, \langle(b)\rangle\}$ permet donc d'affirmer que : $s(\{\langle()\rangle\}) = a(\{\langle()\rangle\}) + a(\{\langle(a)\rangle\}) + a(\{\langle(b)\rangle\}) + a(\{\langle(a)\rangle, \langle(b)\rangle\})$.

Nous savons, d'après la Proposition 5.1, que la matrice inverse contient les différentes valeurs μ nécessaire à la fonction d'inversion de Möbius :

$$\mu = \begin{pmatrix} 1 & -1 & -1 & 1 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Chaque ligne de la matrice μ permet ainsi de calculer l'apparition d'un MCS en fonction des supports de ses sur-motifs. Ainsi, la première ligne contient toutes les valeurs de la fonction de Möbius entre le MCS $\{\langle()\rangle\}$ et tous ses sur-motifs. Ces valeurs permettent alors de calculer $a(\{\langle()\rangle\})$, puisque :

$$\begin{aligned} a(\{\langle()\rangle\}) &= \mu(\{\langle()\rangle\}, \{\langle()\rangle\})s(\{\langle()\rangle\}) + \ldots + \mu(\{\langle()\rangle\}, \{\langle(a)\rangle, \langle(b)\rangle\})s(\{\langle(a)\rangle, \langle(b)\rangle\}) \\ a(\{\langle()\rangle\}) &= s(\{\langle()\rangle\}) - s(\{\langle(a)\rangle\}) - s(\{\langle(b)\rangle\}) + s(\{\langle(a)\rangle, \langle(b)\rangle\}) \end{aligned}$$

Ainsi donc, à partir de la matrice μ, nous pouvons extraire 4 règles de dérivations possibles (puisque par définition la valeur d'une apparition ne peut pas être négative) :

1. $s(\{\langle\rangle\}) - s(\{\langle a\rangle\}) - s(\{\langle b\rangle\}) + s(\{\langle a\rangle, \langle b\rangle\}) \geqslant 0$
2. $s(\{\langle a\rangle\}) - s(\{\langle a\rangle, \langle b\rangle\}) \geqslant 0$
3. $s(\{\langle b\rangle\}) - s(\{\langle a\rangle, \langle b\rangle\}) \geqslant 0$
4. $s(\{\langle a\rangle, \langle b\rangle\}) \geqslant 0$

Avec ces règles et dans le cadre de la base $\mathcal{D}$, nous pouvons alors déduire que $s(C) \in [1, 1]$. Ainsi, le support de ce motif conjonctif séquentiel a été dérivé grâce à ses sous-motifs sans passer par une phase de comptage.

Nous avons présenté dans cette section une approche basée sur le principe d'inversion de Möbius afin de dériver des règles sur le support de motifs conjonctifs. Remarquons que ce principe appliqué aux itemsets permet de retrouver exactement les mêmes résultats au niveau du type de règles de déductions introduits dans [Cal03] et qui sont la base de la représentation condensée par itemsets non-dérivables. Malheureusement, dans notre approche, la monotonie de la dérivabilité dans le cadre des conjonctions de séquence n'est pas assurée ce qui ne permet pas d'optimiser le parcours et la génération de candidats dans l'espace de recherche.

Algorithme 4 : ALGORITHME D'EXTRACTION DE MCS FRÉQUENTS

Data : Une base de données transactionnelles $\mathcal{D}$; un seuil de support minimal σ

Result : L'ensemble de MCS fréquents : $\mathcal{C}$

```
1  begin
2      k ← 1;
3      C ← ∅;
4      F_1 ← mine_sequence(k);
5      C ← F_1;
6      while F_k not empty do
7          foreach s ∈ F_k do
8              foreach x <_{inv prefix} s with x ⋠ s do
9                  C ← generate_CSP(x, s);
10                 Derive_Support(C);
11                 if C not derivable then
12                     Count_Support(C);
13         k++;
14         F_k ← mine_sequence(k);
15     return C;
16 end
```

La prochaine section introduit et discute les différentes étapes de notre algorithme d'extraction de motifs conjonctifs séquentiels.

4 Algorithme

Dans cette section nous introduisons un algorithme de type *profondeur d'abord* pour le problème d'extraction de motifs conjonctifs séquentiels fréquents. L'idée principale de l'algorithme est d'alterner entre une étape d'extraction de motifs séquentiels puis une étape de génération et de comptage des motifs conjonctifs séquentiels candidats.

L'algorithme exploite deux propriétés : (*i*) tout d'abord, afin de compter le support d'un motif conjonctif séquentiel de taille plus grande que 1, il suffit de calculer l'intersection des différents ensembles de transactions de chaque séquence du motif conjonctif séquentiels, (*ii*) dériver pour chaque motif conjonctif séquentiel des règles de supports permettant ainsi de construire une représentation condensée pour les motifs conjonctifs séquentiels.

La structure de données utilisée pour stocker les motifs conjonctifs séquentiels est un arbre préfixé contenant dans chacun de ses nœuds une séquences.

Revenons maintenant sur le calcul de support des motifs conjonctifs. Pour optimiser cette étape de calcul nous nous basons sur une représentation verticale comme pour l'algorithme SPADE [Zak01]. Pour chaque séquence fréquente, nous associons une liste de transactions supportant cette séquence (*tidlist*). Le support d'un motif conjonctif séquentiel est alors le cardinal de l'intersection des différentes *tidlists* des séquences contenues dans le motif conjonctif séquentiel.

Exemple 5.6. Soit la base de données transactionnelles suivante :

$\mathcal{D} =$

Id. Séq	Séquences
S_1	$(a)(b)(c)$
S_2	(a,c)(b)

Supposons que l'on veuille calculer le support du motif conjonctif séquentiel $C = \{\langle(a)(b)\rangle, \langle(a)(c)\rangle\}$.

La *tidlist* de la séquence $\langle(a)(b)\rangle$ est $\{S_1, S_2\}$, celle de la séquence $\langle(a)(c)\rangle$ est $\{S_1\}$, l'intersection de ces deux *tidlists* est : $\{S_1\}$, donc le support du motif conjonctif séquentiel C est 1.

La dérivation de règles nécessite, comme nous l'avons vu dans la section précédente, que tous les sous-motifs soient déjà extraits et leurs supports connus. Cette propriété n'est malheureusement pas respectée avec une approche en profondeur d'abord simple. Afin de résoudre ce problème de parcours de l'espace de recherche nous inversons l'ordre de traverse de l'espace comme présenté dans [CG05a].

Exemple 5.7. Soit la base de données transactionnelles suivante :

$\mathcal{D} =$

Id. Séq	Séquences
S_1	$(a)(b)(c)$

Supposons que le seuil de support minimal est $\sigma = 1$, tout d'abord toutes les séquences de taille 1 sont extraites, puis toutes les conjonctions de ces séquences sont générées en sens inverse. Quand il n'est plus possible de générer de nouvelles conjonctions de séquences candidates, l'algorithme passe alors par une étape d'extraction de séquences fréquentes de taille 2 puis renouvelle l'étape de génération de conjonctions de séquences et de comptage. L'algorithme s'arrête lorsqu'il n'y a plus de séquences fréquentes à extraire.

Le parcours de cet espace et la génération de motifs conjonctifs séquentiels sont illustrés dans les Figures 5.2 et 5.3. Les indices dans chaque nœud de l'arbre représentent l'ordre de parcours.

L'algorithme est présenté dans Algorithme 4.

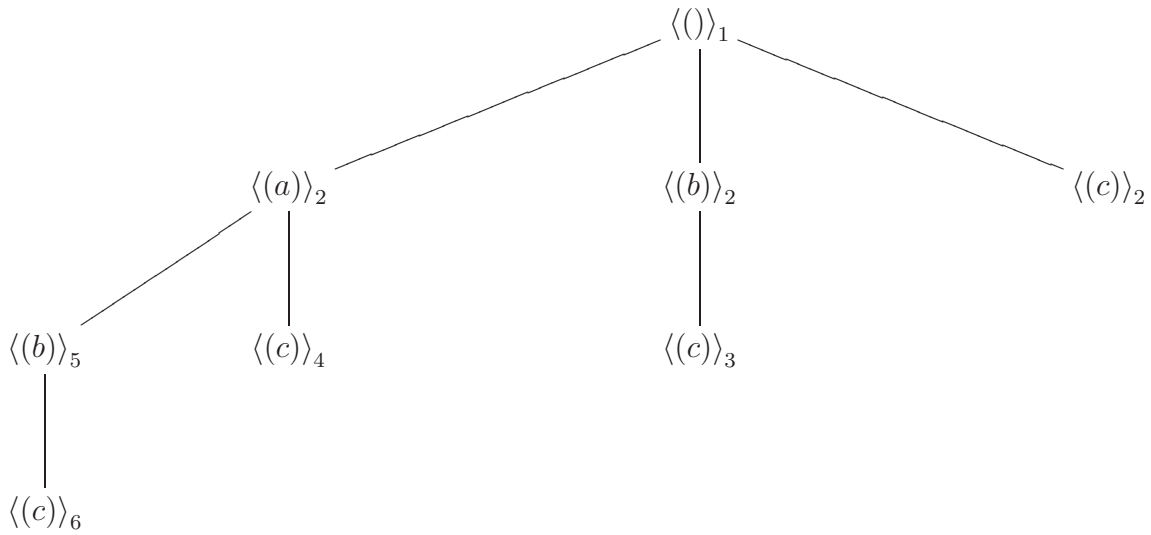


FIG. 5.2: Trace de l'algorithme d'extraction de MCS pour $\mathcal{D} = \{(a)(b)(c)\}$ à l'étape 1

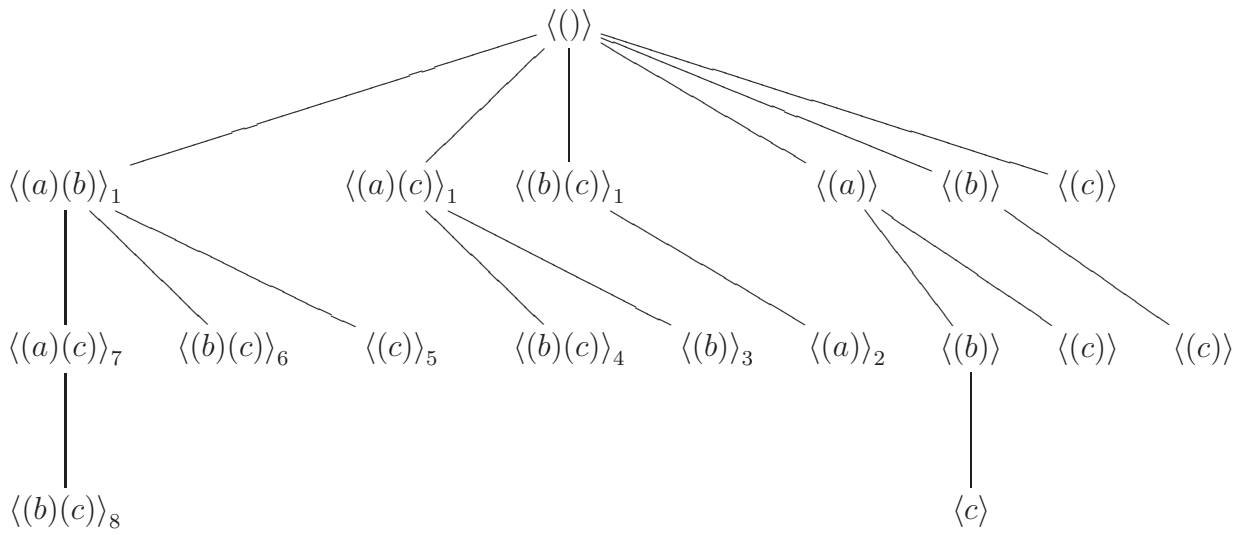


FIG. 5.3: Trace de l'algorithme d'extraction de MCS pour $\mathcal{D} = \{(a)(b)(c)\}$ à l'étape 2

Data set	# de séquences	# d'items	# it / trans.	# trans. / séquence
$C_{200}T_{2.5}S_{10}I_{10K}$	200K	10K	2.5	10
UNIX User Data	11116	2016	1	39

Tab. 5.1: Details des différents jeux de données utilisés dans les expérimentations. # it / trans : nombre moyen d'items par transaction ; # trans. / séquence : nombre moyen de transactions par séquence

5 Expérimentations

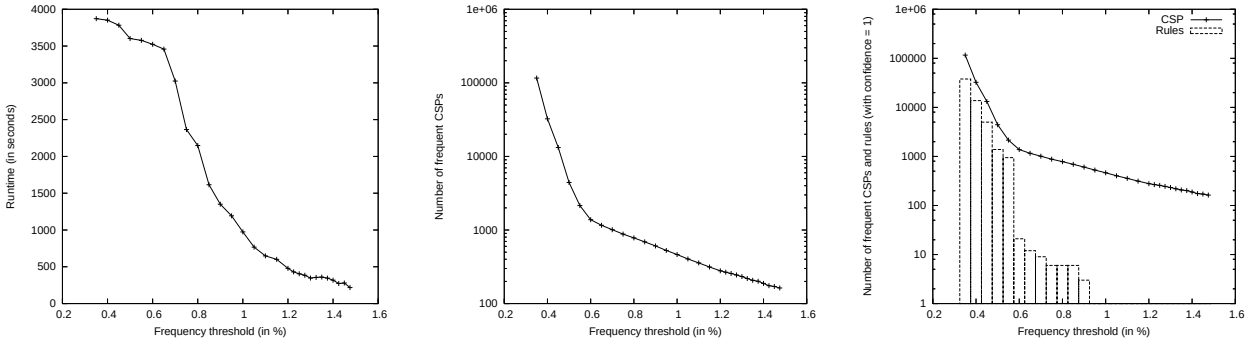
Pour évaluer les performances de notre algorithme d'extraction de motifs conjonctifs séquentiels nous avons réalisé des expériences sur des jeux de données synthétiques et réels. Les premières expérimentations ont pour buts de tester l'extraction de motifs conjonctifs séquentiels et de les comparer avec l'extraction de motifs conjonctifs séquentiels non-dérivables. Le reste des expérimentations a pour but de comparer notre algorithme avec une approche naïve d'extraction de motifs conjonctifs séquentiels.

5.1 Méthode expérimentale

Toutes les expérimentations ont été lancées sur un ordinateur MacBookPro Core-Duo 2.16 Ghz doté de 2Go de mémoire, tournant sous Mac OS X 10.5.2. L'algorithme d'extraction est implementé en C++ en se basant sur la librairie [1] et les structures internes de l'implémentation de SPADE. Nous avons utilisé deux jeux de données : un jeu de données synthétiques généré grâce à l'outil de génération de jeux de données QUEST[2] et un jeu de données réels contenant les logs de 8 ordinateurs sous systèmes UNIX de l'université de Purdue (le jeu de données a été construit sur une durée de 2 ans).

Le jeu de données synthétiques $C_{200}T_{2.5}S_{10}I_{10K}$ contient 200000 séquences basées sur 10000 items. Le jeu de données *UNIX User Data*, quant à lui, contient 9 ensembles de commandes d'utilisateurs distribués sur 8 ordinateurs pour un total de 11116 séquences. Il peut être très intéressant d'extraire des règles d'association séquentielles sur ce jeu de données pour les utiliser plus tard dans un système de détections d'intrusions.

Les détails de ces 2 jeux de données sont données dans la Table 5.1.



(a) Temps **(b)** Nombre de MCS **(c)** Nombre de MCS et règles extraites

FIG. 5.4: Expérimentations sur le jeu de données $C_{200}T_{2.5}S_{10}I_{10K}$

5.2 Résultats

Jeu de données synthétiques. Faisabilité et validité.

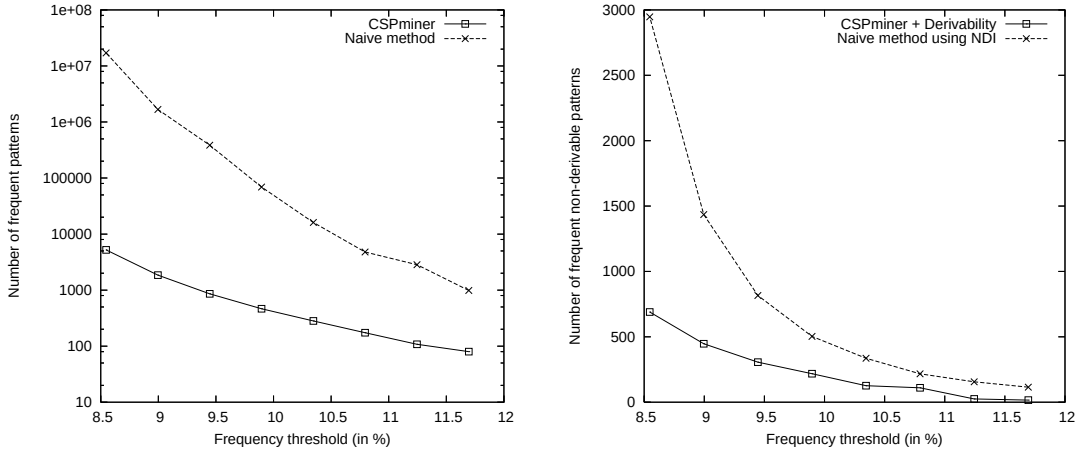
Les premières expérimentations que nous avons faites visaient à étudier les effets de la valeur de support minimal sur le nombre de MCD extraits, le temps nécessaire pour l'extraction et le nombre de règles d'association séquentielles que l'on peut extraire de ces MCS pour une valeur de confiance de 1. Les effets d'un seuil de support minimal bas sont présentés dans la Figure 5.4(a). Le temps d'extraction est quand à lui très acceptable puisque pour un support de 1.15%, notre algorithme a besoin de moins de 10 minutes pour compléter l'extraction des MCS. De plus, notre méthode semble assez résistante à des seuils de supports très bas (en terme de complétion) puisque pour le jeu de données $C_{200}T_{2.5}S_{10}I_{10K}$, nous avons pu descendre jusqu'à un seuil de 0.35% avec un temps de traitement avoisinant 1 heure et 4 minutes. La Figure 5.4(b) illustre la capacité de l'algorithme à extraire un très grand nombre de MCS pour différentes valeurs de seuil de support. Comme on peut le voir pour un seuil de 0.35%, la structure de données contient à peu près 100000 motifs. Le nombre de règles d'association séquentielles avec une confiance de 1 par rapport au nombre de motifs extraits sont représentés dans la Figure 5.4(c). Remarquons qu'il n'y a pas de règles extraites jusqu'à un seuil de support de 0.95%. Ceci s'explique par le fait que jusqu'à ce seuil, les seuls motifs extraits sont des séquences de taille 1 qui ne peuvent pas être utilisées pour extraire les règles.

UNIX User Data set et non-dérivabilité.

Nous avons utilisé le jeu de données *UNIX User Data* afin de souligner l'utilité d'une représentation condensées basée sur le concept de non-dérivabilité. Nous comparons ainsi notre algorithme avec une approche naïve consistant en une étape de post-traitement après l'extraction de tous

[1]http ://sourceforge.net/projects/dmtl

[2]http ://www.almaden.ibm.com/cs/projects/iis/hdb/Projects/data_mining/



(a) MCS **(b)** MCS non-dérivable

Fig. 5.5: Expérimentations sur le jeu de données *UNIX User Data*.

les motifs séquentiels. A chaque séquence nous associons un item et le jeu de données est transformé afin de permettre ainsi l'extraction d'itemsets fréquents. Nous avons procédé à deux types d'extraction d'itemsets fréquents : (i) une extraction classique de tous les itemsets fréquents (en utilisant l'algorithme *Apriori*) et (ii) une extraction d'itemsets fréquents non-dérivables[1]. La Figure 5.5(a) illustre la différence entre le nombre de MCS extraits par notre approche et l'approche naïve. Comme on peut le voir, il existe une grosse différence sur le nombre de MCS extraits par ces deux méthodes. A un seuil de 8.54%, notre algorithme extrait 5211 MCS alors que l'approche naïve extrait plus de 17 millions de conjonctions de séquences. Ceci s'explique par le fait que la méthode naïve ne fait pas de comparaisons entre les séquences présentes dans les conjonctions, alors que notre algorithme n'extrait que les motifs conjonctifs séquentiels selon la Définition 5.1. Par exemple, la conjonction de séquence $\{\langle(a)(b)\rangle, \langle(a)\rangle\}$ sera extraite par l'approche naïve alors qu'elle ne sera même pas générée par notre approche car $\langle(a)\rangle \preceq \langle(a)(b)\rangle$.

Les tests sur la non-derivabilité sont présentés dans la Figure 5.5(b) et la Table 5.2. Comme on pouvait l'espérer par rapport à l'approche naïve, notre algorithme montre un grand ratio de conjonctions dont le support peut être dérivé. Les ratios présentés dans la Table 5.2 soulignent l'utilité d'une telle représentation condensée pour ces nouveaux motifs. Néanmoins, on peut remarquer que la méthode naïve a de meilleurs ratios que notre approche : ceci est probablement dû au nombre de conjonctions générées et extraites par l'approche naïve (plus d'un million de conjonctions avec un support de 9%).

[1]Les implémentations ont été téléchargés à l'adresse web suivante http ://www.adrem.ua.ac.be/ goethals/software/

Support	# MCS	# Conj.	# NDMCS	# NDconj.	ratio MCS	ration Conj.
11.69%	80	984	16	115	20%	88.31%
11.24%	108	2840	25	156	23.14%	94.50%
10.79%	174	4784	110	217	36.78%	95.46%
10.34%	282	16146	126	336	44.68%	97.91%
9.89%	466	68991	218	503	53.21%	99.27%
9.44%	858	385107	306	816	64.29%	99.78%
8.99%	1856	1677387	447	1435	75.92%	99.91%
8.54%	5211	17127924	690	2947	86.75%	99.98%

TAB. 5.2: Details du jeu de données *UNIX User Data* et des différends ratios pour notre approche et la méthode naïve # MCS : nombre de MCS extraits ; # Conj. : Nombre de conjonctions fréquentes extraites (méthode naïve).# NDMCS : nombre de MCS non-dérivables. # NDconj. : Nombre de conjonctions fréquentes non-dérivables (méthode naïve)

6 Discussions

Nous avons introduit dans ce chapitre un nouveau motif basé sur les séquences. Ainsi, nous avons présenté les différentes définitions formelles associées à ce motif. Contrairement au problème d'extraction de motifs séquentiels, nous avons montré que dans le cas des motifs conjonctifs séquentiels il était possible de dériver des règles de support sur ces nouveaux motifs permettant ainsi de construire une représentation condensée similaire à la représentation par itemsets fréquents non-dérivables. De plus, ce nouveau motif est d'un grand intérêt dans l'optique d'extraction de règles d'associations séquentielles.

A notre connaissance, seules les travaux de G. C. Garriga [CG06, BCG07] avaient essayé de formaliser l'approche d'extraction de règles d'association séquentielles. Ses approches se basent sur une formalisation complète de la théorie des treillis pour les séquences avec un ensemble d'opérateurs de fermetures permettant d'extraire uniquement les conjonctions de séquences fermés. Les règles sont alors extraites à partir d'une autre formalisation en se basant sur les clauses de Horn. Cette approche bien que très intéressante d'un point de vue théorique, est quelque peu limitée d'un point de vue pratique puisqu'elle nécessite une approche en post-traitement après l'extraction de motifs séquentiels clos. De plus, la formalisation par clauses de Horn ne permet pas d'avoir des règles d'associations généralisées puisque dans l'approche de G. C. Garriga, une seule séquence peut être dans le prémice de la règle alors que dans notre approche il peut y avoir une conjonction de séquences en prémice et en conclusion.

Nous pensons que les travaux présentés dans ce chapitre, permettront dans un avenir proche d'extraire des règles d'associations pour des motifs plus complexes tel que les épisodes, les arbres et les graphes. L'approche des classes d'équivalences sur la relation de support et la construction de conjonctions pourraient alors être étendues à ces motifs afin de permettre une meilleure compréhension et optimisation des techniques de fouilles de ces motifs. Ainsi des règles d'associations sur les graphes seraient du plus grand intérêt dans les domaines bioinformatique, biologique, médicale ou bio-chimique.

Deuxième partie

Extraction de Motifs Séquentiels sur les flots de données

Citation
Auteur (Date) — *Source*

Dans la partie précédente, nous nous sommes intéressés aux motifs séquentiels dans le cadre de bases de données statiques. L'objectif de cette partie est de considérer un nouveau modèle de données : les flots. En effet, comme nous l'avons vu en introduction, avec le développement de nouvelles technologies, les applications sont de plus en plus obligées de faire face à des données disponibles en temps réel, de manière continue et ordonnées. Bien entendu, les approches d'extraction doivent être reconsidérées dans un tel contexte car elles ne sont pas définies pour aborder de tels modèles. Pour s'en convaincre, considérons le cas des algorithmes d'extraction d'itemsets utilisant une stratégie de type générer-élaguer. Le principe de ces approches, comme nous l'avons vu précédemment est de générer à partir de données fréquentes des ensembles de candidats et de parcourir la base de données afin de calculer la fréquence d'apparition d'un candidat afin de le conserver ou non. Nous voici donc confronté au problème : la base de données n'est pas disponible dans le cas de flots de données. Même si nous prenons des approches basées sur des projections, nous sommes confrontés au même problème : il faut au moins une fois accéder à l'intégralité de la base de données.

Dans cette partie nous reconsidérons la problématique d'extraction de motifs séquentiels dans le cadre de flots de données. Après avoir présenté un aperçu des modèles et des approches d'extraction existantes dans le Chapitre 6, nous répondrons dans le Chapitre 7 à la question suivante : *est-il possible d'utiliser une approche par échantillonnage pour extraire des motifs séquentiels dans des flots de données ?* L'objectif du chapitre 8 est d'étendre le pouvoir d'expression des motifs en intégrant les différentes dimensions associées aux données. Nous souhaitons ainsi extraire des motifs séquentiels multidimensionnels dans des flots.

Chapitre 6

Les flots de données

L'objectif de ce chapitre est de présenter la problématique de l'extraction de motifs sur les flots de données. Dans la section 1, nous présentons les différents modèles et définitions des flots de données. La section 2 présente quant à elle les différents travaux de recherches existants dans le cadre de l'extraction de connaissances sur les flots. Nous conclurons ce chapitre par une discussion.

1 Modèles et définitions

Ces dernières années, avec le grand *boom* des nouvelles technologies dans le domaine des logiciels et matériels informatiques, de nouvelles applications sont apparues, générant une quantité de données jamais atteinte dans l'histoire de l'humanité. Ainsi, les opérations de stockage et d'analyse de ces données deviennent tout simplement impossibles à réaliser sans l'apport de nouveaux modèles et algorithmes. C'est ainsi qu'est apparu le modèle de *flot de données*.

Un flot de données est une séquence de données, structurées ou non, potentiellement infinie et volatile. Par volatile, nous désignons la capacité de la séquence de données à être générée de manière très rapide et son impossibilité à être stockée entièrement.

Il existe plusieurs domaines d'applications des flots de données, citons notamment les applications :

1. De télécommunications, comme les réseaux TCP/IP, qui peuvent être vus comme des flots de données ou chaque machine connectée génère un ensemble souvent impressionnant de données (de l'ordre de quelques giga-octets par jour et par machine). Ces applications peuvent aller de la détection de *Déni de Service* et des *Heavy-Hitters* à l'analyse des points d'engorgement réseaux et des anomalies.

2. Financières. En effet, il est tout fait naturel d'imaginer que les cours des valeurs et indices financiers peuvent être modélisé comme un flot de données. Les applications pour l'utilisateur sont nombreuses comme la détection de tendances, l'extraction des k valeurs les plus intéressantes (avec une mesure d'intérêt) et la découverte de règles de corrélations entre les indices (par exemple : lorsque l'indice A perd 2 points l'indice B prend toujours 4 points).

3. De capteurs. Avec la montée en puissance des capteurs de tous types, il devient de plus en plus nécessaire d'analyser le volume de données faramineux généré par ces réseaux. Les applications sont généralement issues du domaine de la météorologie ou de la sismologie.

Modèles de flots

Il existe trois modèles de flots de données [Agg07]. Le choix du modèle dépend bien sûr de l'application finale et des besoins de l'utilisateur. Dans le reste de ce chapitre nous suivrons les notations et définitions de [Mut03]. Notons un flot de donnée F. Les élément arrivant sur ce flot sont notés $a_1, a_2, \ldots$ et décrivent un signal sous-jacent à p variables $A : [0, \ldots, p-1]$ à valeurs dans $\mathbb{R}$. Les trois modèles suivants diffèrent uniquement sur la manière dont les éléments a_i décrivent le signal A :

– Le modèle des *séries temporelles* : dans ce modèle chaque élément $a_i = A[i]$, il existe donc une adéquation complète entre le flot de données et le signal qu'il décrit.

– Le modèle de *la caisse enregistreuse* : dans ce modèle chaque élément a_i vient s'ajouter (ou s'incrémenter) à $A[j]$. En fait, on peut définir chaque élément $a_i = (j, I_i)$ avec I_i positif et $A_i[j] = A_{i-1}[j] + I_i$. Sémantiquement parlant, ce modèle tient compte de l'augmentation du signal pour une valeur $j \in [0, \ldots, p-1]$ à chaque instant i. Ce modèle est le modèle le plus utilisé généralement dans les applications de flots de données.

– Le modèle *du tourniquet* : dans ce modèle, et contrairement au modèle précédent de la caisse enregistreuse, chaque élément a_i peut soit s'ajouter soit se soustraire à la valeur précédente dans $A[j]$. Ainsi, pour chaque élément $a_i = (j, I_i)$ avec I_i positif ou négatif, $A_i[j] = A_{i-1}[j] + I_i$. Bien que ce modèle soit le plus général possible, il est très rarement utilisé par les applications. Ceci est dû généralement à la difficulté de calculer des bornes sur la variation du signal A.

Gestion du temps

Un des aspects primordiaux d'un flot de données est le caractère temporel. Les éléments arrivant sur le flot possèdent chacun une estampille temporelle. Etant donné qu'un flot de données est infini, il est généralement impossible, matériellement parlant, de stocker et de traiter tous les éléments d'un flot. Il est donc nécessaire de restreindre notre traitement à une portion du flot. Cette portion est généralement appelée fenêtre [Agg07, Mut03, BDM02]. Il existe plusieurs type de fenêtres, nous allons ici décrire les plus importantes d'entre elles :

- La fenêtre fixe. Cette fenêtre est une portion stricte du flot de données. Par exemple la fenêtre stricte entre les données du début du mois d'Avril jusqu'à la fin du mois de Mai.

- La fenêtre glissante. Cette fenêtre contient deux bornes dynamiques mises à jours selon deux conditions possibles : soit par réévaluation après chaque nouvelle arrivée d'un élément sur le flot (réévaluation avide), soit la réévaluation se fait après une période de temps bien précise choisie par l'utilisateur.

- Les tables de fenêtres naturelles ou logarithmiques [CDH+02, GHP+03]. Les tables de fenêtres se basent généralement sur une représentation des éléments du flot à différents niveaux de granularité temporelle. Cette notion a initialement été introduite dans [CDH+02] et est basée sur l'intuition suivante : les personnes sont plus souvent intéressées par les récents changements avec une granularité fine et par les changements à long terme avec une granularité plus large. De manière à illustrer plus longuement cette notion, nous reportons ici l'exemple tiré de [GHP+03]. La Figure 6.1 illustre une table de fenêtres naturelles : les 4 quarts d'heure plus récents, puis les dernières 24 heures et enfin 31 jours. Lorsque les 4 valeurs du premier niveau (les quarts d'heures) sont remplies, nous regroupons ces valeurs dans le niveau de granularité temporelle suivant, c'est-à-dire le niveau de granularité des heures et ainsi de suite jusqu'au dernier niveau de granularité possible. A partir de ce modèle, nous pouvons savoir ce qui s'est passé la dernière heure avec une précision d'un quart d'heure, le dernier jour avec une précision d'une heure, etc. Ainsi, il est possible pour chaque élément apparaissant sur le flot de garder un historique bien précis de ces apparitions et selon les besoins de l'application ou l'utilisateur final. Dans [GHP+03], les auteurs proposent également d'étendre ces tables de fenêtres naturelles en tables de fenêtres logarithmiques. L'avantage est d'avoir une représentation temporelle beaucoup plus compacte en introduisant en contrepartie un mécanisme d'approximation. Les tables de fenêtre logarithmiques sont mises à jour par effet de cascade comme illustré dans la Figure 6.2, lorsque le niveau de granularité le plus bas est rempli, ici le niveau 0 contenant 2 valeurs, nous regroupons ces valeurs vers le

niveau de granularité suivant (le niveau 1) et procédons au même regroupement si ce niveau contient plus de 2 valeurs et ainsi de suite jusqu'au dernier niveau.

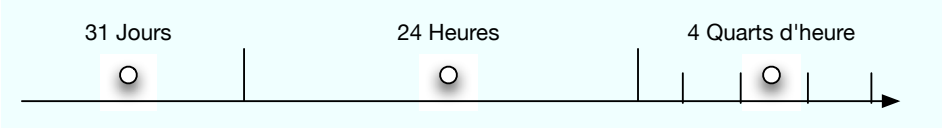


Fig. 6.1: Exemple de table de fenêtres naturelles

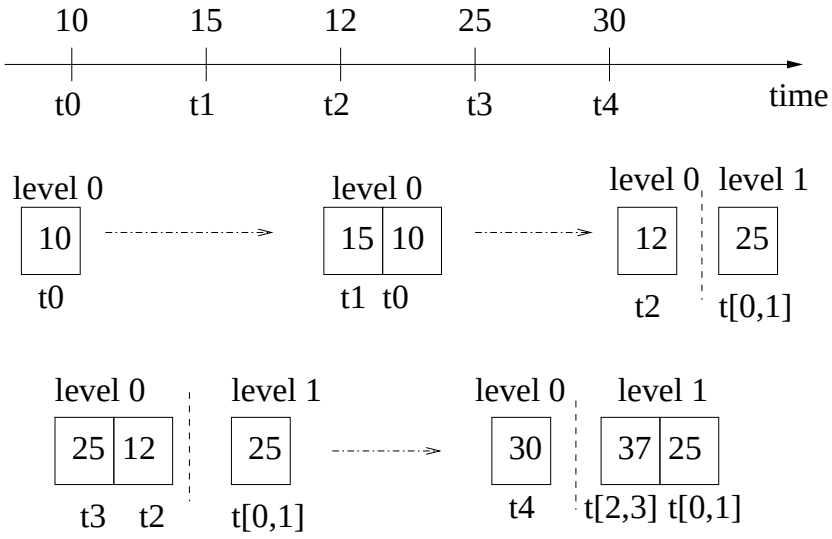


Fig. 6.2: Exemple de table de fenêtres logarithmiques

2 Extraction de connaissances sur les flots

L'extraction de connaissances sur les flots de données est devenue un enjeu majeur pour les entreprises et les organisations. Ainsi, pouvoir extraire de la connaissance en temps réel, permet à l'analyste, ou au décideur, de prendre en compte très vite les changements brusques apparus dans ses données lui donnant ainsi de meilleurs options dans sa prise de décision finale. La plupart des problématiques classiques de l'extraction de fouille se retrouve dans les algorithmes de fouille de flot de données comme par exemple : la segmentation, la classification (par arbres de décisions) ou l'extraction de motifs fréquents [Agg07, DH00]. La seule différence est que les algorithmes classiques permettent généralement plusieurs passes alors que dans le cas des flots de données seule une passe est possible sur les données. Il n'est généralement donc pas possible d'adapter

ces algorithmes et il devient nécessaire de trouver de nouvelles méthodes radicalement différentes pour l'extraction de connaissances sur les flots. Enfin, une dernière difficulté est la rapidité de réponse des algorithmes qui doivent pouvoir répondre aux requêtes ou compléter une extraction avant l'arrivée des prochaines informations du flot.

2.1 Extraction d'itemsets

Dans cette sous-section, nous nous focalisons plus particulièrement sur les travaux réalisés autour de la recherche d'itemsets fréquents dans les flots de données. Cette présentation a deux avantages. D'une part, la recherche d'itemsets peut en fait être perçue comme un cas particulier de séquences réduites à un seul itemset. D'autre part elle nous permet de présenter quelques aspects qui sont, ou peuvent être, réutilisés dans le contexte d'extraction de séquences sur les flots.

La première approche d'extraction d'itemsets dans des flots de données a été proposée par Manku et Motwani [MM02]. Les auteurs proposent un algorithme, le *Lossy Counting Algorithm* en une passe basé sur un partage du flot entier en *buckets*. Cet algorithme permet ainsi d'approximer l'ensemble d'itemsets fréquents sur le flot. Pour cela, les auteurs utilisent une structure à base d'arbres pour représenter les itemsets et la taille des *buckets* est choisie selon un taux d'erreur ϵ. La taille de ces *buckets* et la mise à jour du comptage se base sur le concept de ϵ-*deficient function*. Un des mérites de cette approche est qu'il ne peut pas y avoir de faux négatifs et l'erreur sur la fréquence des itemsets est garantie de ne pas dépasser un certain seuil.

Li et al. [LLS04] utilisent une extension d'une représentation basée sur un ensemble d'arbres (*Item-Suffix Frequent Itemset Forest*) dans leur algorithme $DSM\text{-}FI$. Cet algorithme propose aussi une approximation de l'ensemble des itemsets fréquents. Ainsi, dans leur approche et comme pour le *Lossy Counting Algorithm*, les auteurs proposent de relaxer le seuil de fréquence minimal permettant ainsi de mieux garantir la précision de l'extraction. Tous les sous-itemsets extraits sont stockés dans une forêt d'arbres suffixés et de tables d'en-têtes. Les auteurs soulignent que cette représentation par forêt est plus compacte que la représentation par arbre préfixé, malheureusement ce gain en espace mémoire se paye par un plus grand temps de calcul.

Yu et al. [YCLZ04] proposent l'algorithme $FDPM$ qui se base sur les inégalités de concentration et notamment *l'inégalité de Chernoff* (cf. Chapitre 7) qui permet aux auteurs de borner les fréquences des itemsets fréquents selon une (ϵ, δ)-approximation, ainsi, la probabilité que le support d'un itemset I dans les n premières transactions d'un flot diffère du support global de l'itemset I de plus ou moins ϵ ne dépasse pas les $(1 - \delta)$. Cette approche, basée sur des calculs de

bornes statistiques permet d'éviter les faux-positifs dans le cadre de l'extraction.

Les auteurs de [CL03], quand à eux, proposent de ne travailler que sur la partie récente du flot et introduisent ainsi une approche basée sur le *taux d'oubli* afin de limiter et de diminuer au fur et à mesure du temps le poids des anciennes transactions. Ainsi l'algorithme permet de se concentrer uniquement sur les parties récentes du flot, mais ce principe *d'oubli* introduit un trop grand taux d'erreur sur la fréquence et peut s'avérer difficile à gérer dans le cas d'applications critiques.

D'autres auteurs préfèrent se pencher sur des algorithmes afin d'extraire des représentations condensées comme Chi et al. [CWYM04] qui considèrent la recherche des itemsets fermés fréquents dans l'algorithme *Moment*. Pour cela, ils proposent un arbre d'énumération des fermés (*Closed Enumeration Tree - CET*) pour maintenir au cours du temps un ensemble d'itemsets sélectionnés : les itemsets clos et ceux pouvant devenir potentiellement clos. L'algorithme *Moment* permet ainsi d'extraire de manière précise les itemsets clos sur une fenêtre glissante du flot, malheureusement, si la fenêtre glissante est trop grande, la structure d'arbre d'énumération des itemsets fermés risque d'être très gourmande en mémoire et limitera ainsi les performances de mises à jours.

Finalement, dans [GHP+03], les auteurs proposent d'utiliser une structure de type FP-tree [HPMa+00] pour rechercher des itemsets fréquents à différents niveaux de granularité temporelle. Pour cela, ils introduisent la technique des *tilted-time windows table* (les tables de fenêtres temporelles logarithmiques). Cette notion a initialement été introduite dans [CDH+02] et est basée sur l'intuition suivante : les personnes sont plus souvent intéressées par les récents changements avec une granularité fine et par les changements à long terme avec une granularité plus large. L'utilisation de fenêtre temporelles logarithmiques dans cet algorithme permet ainsi de répondre à des requêtes plus complexes sur différentes granularités et sur différents laps de temps.

2.2 Extraction de séquences

Contrairement à la problématique de fouille d'itemsets sur les flots de données, très peu d'algorithmes ont été proposés pour l'extraction de séquences fréquentes sur les flots de données.

A notre connaissance il existe trois approches de fouilles de données que nous détaillons :

Dans [MM06], Marascu et al. proposent l'algorithme $SMDS$ (*Sequence Mining in Data Streams*) qui permet l'extraction de motifs séquentiels sur les flots. L'algorithme $SMDS$ se base sur une approche par segmentation (*clustering*) de séquences. Les séquences sont regroupées selon une

méthode d'alignement issue de [KPWD03] et permet ainsi de construire au fur et à mesure de l'avancée du flot des ensembles de séquences similaires.

Dans [RPT06], nous avons proposé un algorithme d'extraction de motifs séquentiels maximaux sur les flots de données appelé *SPEED*. L'idée principale est de toujours maintenir une bordure qui contiendrait les séquences maximales sous-fréquentes potentielles et qui pourraient devenir fréquentes au fur et à mesure de l'évolution du flot. Notre approche se base comme dans [GHP+03] sur l'utilisation de tables de fenêtres temporelles logarithmiques afin de pouvoir garder un historique étendu des motifs séquentiels maximaux. L'algorithme *SPEED* procède à une transformation des itemsets afin de représenter chacun des itemsets au moyen d'un seul entier. Pour ce faire, nous associons à chaque item un nombre premier, cette association permet ainsi d'utiliser des propriétés de congruences et d'arithmétiques simples pour les tests d'inclusions de séquences.

La troisième approche d'extraction de motifs séquentiels est quelque peu différente des deux précédentes puisqu'elle prend en compte l'extraction sur un ensemble de flots de données. Les auteurs Chen et al. [CWZ05] considèrent ainsi que les motifs extraits de cet ensemble de flots de données sont des motifs multidimensionnels. Les auteurs choisissent un modèle de fenêtre glissante avec un système d'alignement des transactions des différents flots de données afin d'extraire ces motifs séquentiels multidimensionnels. Leur algorithme, nommé $MILE$, utilise l'algorithme PrefixSpan à chaque étape de glissement afin d'extraire ces motifs.

3 Discussion

Dans ce chapitre, nous avons dans un premier temps, introduit et discuté les différents modèles de flots et de gestion temporelle des flots, puis nous avons discuté des différentes méthodes et algorithmes pour l'extraction d'itemsets fréquents et de motifs séquentiels. Les Tableaux 6.1 et 6.2 décrivent les différentes approches pour l'extraction d'itemsets fréquents et de motifs séquentiels selon les critères suivants : (i) le modèle de gestion temporelle (quel type de fenêtre ?), (ii) Le type de mise-à-jour, est-ce que la structure de données est actualisée à chaque nouvel élément ou après avoir collecté un ensemble de nouveaux éléments (l'ensemble de nouveaux éléments sur le flot est appelé batch) ? (iii) Quel type d'approximation avons-nous ? (iv) Si l'extraction a pour résultat l'ensemble des motifs fréquents ou une représentation condensée.

Comme nous avons pu le voir dans la seconde partie de ce chapitre, il existe très peu d'approches pour l'extraction de motifs séquentiels sur les flots de données. Les quelques approches existantes

	Fenêtre	mise-à-jour	Approx.	Rep. Condensée
Manku et Motwani [MM02]	Tout le flot	Batch	Faux-positifs	Non
Li et al. [LLS04]	Tout le flot	Par batch	Faux-positifs	Non
Yu et al. [YCLZ04]	Tout le flot	Par batch	Faux-négatifs	Non
Chang et al. [CL03]	Fenêtre glissante	Par transaction	Faux-positifs	Non
Chi et al. [CWYM04]	Fenêtre glissante	Par transaction	Exacte	Clos
Gianella et al. [GHP+03]	Table de Fenêtre Log.	Par batch	Faux-positifs	Non

Tab. 6.1: Synthèse des approches pour l'extraction d'itemsets fréquents sur les flots

	Fenêtre	mise-à-jour	Approx.	Rep. Condensée
Marascu et al. [MM06]	Table de Fenêtre Log.	Batch	Faux-positifs	Non
Raïssi et al. [RPT06]	Table de Fenêtre Log.	Par batch	Faux-positifs	Maximaux
Chen et al. [YCLZ04]	Fenêtre glissante	Par transaction	Exact	Non

Tab. 6.2: Synthèse des approches pour l'extraction d'itemsets fréquents sur les flots

n'utilisent en aucun cas les principes d'approximations et d'échantillonnage largement utilisés dans les approches d'extraction d'itemsets. Remarquons aussi qu'aucune approche, que ce soit dans le cadre des itemsets ou des motifs séquentiels n'essaye de prendre en compte la multidimensionnalité inhérente du flot.

Tout au long des prochains chapitres, nous présenterons des approches permettant un échantillonnage efficace sur les flots de données pour une extraction efficace de motifs séquentiels ainsi qu'une approche pour extraire des motifs séquentiels réellement multidimensionnels.

Chapitre 7

Motifs séquentiels et échantillonnage

Nous avons vu précédemment que la prise en compte des contraintes associées aux flots nécessitait de reconsidérer les algorithmes d'extraction de motifs séquentiels qui n'étaient plus adaptés à la nouvelle problématique. De nombreux travaux de recherche ont montré qu'il était aujourd'hui tout à fait acceptable et raisonnable d'avoir une approche approximative plutôt que complète sur les résultats de connaissances extraites du flot. En effet, étant donnée la vitesse à laquelle les données sont disponibles, l'utilisateur final souhaite avoir plutôt un aperçu des tendances (des séquences sont récurrentes, des séquences apparaissent ou disparaissent, ...) ou d'événements particuliers (*e.g.* Top-k) qui surviennent sur le flot. C'est dans cette philosophie que se situe ce chapitre où nous présentons les travaux que nous avons menés dans le cadre de l'échantillonnage pour l'extraction de motifs séquentiels. Une partie des travaux présentés dans ce chapitre a été publiée dans les conférences *IEEE International Conference on Data Mining (ICDM 2007)* et *Extraction et Gestion de Connaissances (EGC 2008)*.

1 Introduction

Nous avons vu dans le chapitre 3, que la problématique de l'extraction de motifs séquentiels dans de grandes bases de données intéresse la communauté fouille de données depuis une dizaine d'années et différentes méthodes ont été ainsi développées pour extraire des séquences fréquentes. L'extraction de tels motifs est toutefois une tâche difficile car l'espace de recherche considéré est très grand. Par exemple, si nous considérons i items, il existe $O(i^k)$ séquences fréquentes possibles de taille k [Zak01].

De nombreuses approches ont été proposées ces dernières années pour gérer au mieux cet espace de recherche (*e.g.* PrefixSpan [PHMa+01], SPADE [Zak01], SPAM [AFGY02]). Les plus tradition-

nelles utilisent une approche à la *Apriori* [SA96] et diffèrent principalement par les structures de données utilisées (vecteurs de bits, arbres préfixés, ...). Les approches les plus récentes considèrent, quant à elles, des projections multiples de la base de données selon le principe de *pattern-growth* proposé dans [PHMa+01] et évitent ainsi l'étape de génération de candidats. Cependant même si ces stratégies sont efficaces, les approches existantes[1] considèrent une hypothèse forte : la base de données peut être chargée directement en mémoire centrale. Par exemple, PrefixSpan, basé sur la représentation de motifs au moyen d'un arbre préfixé, considère dans son implémentation publique que la base de données est chargée en mémoire centrale et utilise différentes projections afin de minimiser la génération des candidats. SPADE propose, par contre, de transformer l'intégralité de la base de données en une représentation verticale (items-clients) afin d'appliquer rapidement des opérations de jointure. Enfin, dans le cas de SPAM l'originalité réside dans la représentation de l'intégralité de la base de données sous la forme de bitmaps.

Malgré le développement des nouvelles capacités de stockage, ces approches se trouvent, cependant, de plus en plus mises en défaut dans la mesure où la quantité de données manipulées est trop volumineuse et qu'il devient irréaliste d'essayer de stocker l'intégralité de la base en mémoire centrale. En d'autres termes, nous nous retrouvons aujourd'hui confronté aux problèmes suivants :

- *Est-il possible de trouver un échantillon représentatif de la base de données d'origine qui puisse tenir en mémoire centrale et pour lequel nous sommes capables de maîtriser la marge d'erreur sur les résultats de l'extraction ?*
- *Cet échantillonnage est-il possible tout en respectant la relation d'ordre induite par les séquences ?*

Si nous considérons, à présent, les flots de données, le fait de devoir faire plusieurs parcours sur la base ou d'y avoir un accès intégral fait que les approches traditionnelles d'extraction ne sont plus adaptées. Au cours de ce chapitre, nous souhaitons que notre approche soit capable de répondre aux nouvelles contraintes et également répondre aux questions suivantes :

- *Est-il possible d'appliquer une technique d'échantillonnage sur le flot pour l'extraction de motifs séquentiels ?*
- *Sommes-nous à même de garantir que les résultats d'une extraction de motifs sont réellement représentatifs du contenu du flot*
- *Pouvons-nous garantir une marge d'erreur fixe lors de cet échantillonnage ?*

[1]Les différentes implémentations peuvent être téléchargés sur les sites webs suivants : http ://dmtl.sourceforge.net/ et http ://illimine.cs.uiuc.edu/

Au cours de ce chapitre, nous répondons à toutes ces questions. En partant du cas d'une base de données statique, nous montrons comment minimiser le nombre d'accès, en terme d'entrées-sorties, afin d'extraire une approximation contrôlée des motifs séquentiels. Cette approche est basée sur la construction d'un échantillon aléatoire issu du jeu de données original. Nous répondons à la question théorique de la précision de l'échantillon de la base en bornant la taille de l'échantillon par rapport à un seuil d'erreur choisi par l'utilisateur. D'autre part, nous proposons une extension des résultats de l'échantillonnage pour les bases de données statiques afin de prendre en compte le modèle de flot de données et répondons ainsi aux différentes questions soulevées. Nous proposons ainsi une nouvelle méthode de pré-traitement des données afin de faciliter l'extraction des motifs séquentiels. Ce pré-traitement correspond en fait à un algorithme de maintien de résumés basé sur les approches d'échantillonnage par réservoir [Vit85].

Le chapitre est organisé de la manière suivante. Les concepts préliminaires sont proposés dans la section 2. Dans la section 3, nous introduisons la problématique de l'échantillonnage de motifs séquentiels dans les bases de données statiques. La section 4 étend les résultats obtenus précédemment afin de les transposer dans le cadre de l'échantillonnage pour l'extraction de motifs séquentiels dans les flots de données. Les expériences réalisées sont décrites et discutées dans la section 5. Nous concluons ce chapitre par une discussion.

2 Concepts Préliminaires

Au cours de cette section, nous présentons tout d'abord les principes de l'échantillonnage par réservoir et montrons que les approches récentes proposées par Aggarwal [Agg06] offrent des mécanismes tout à fait adaptés à la prise en compte des flots de données. Par la suite nous présentons un outil statistique fréquemment utilisé afin de borner les différentes erreurs introduites par l'échantillonnage : les inégalités de concentrations.

2.1 Échantillonnage par réservoir

De nombreuses structures de synopsis ont été développées ces dernières années comme les sketches, l'échantillonnage, les ondelettes et les histogrammes. Toutes ces différentes approches possèdent les mêmes propriétés de grandes applicabilités (elles peuvent être utilisées pour répondre à divers problèmes), d'efficacité mémoire (elles sont capables de résumer de manière significative de grandes quantité de données) et de robustesse. En outre, toutes ces méthodes ne nécessitent généralement qu'une passe sur les données à échantillonner ce qui les rend particulièrement utiles dans le cas des flots de données.

Dans les travaux que nous avons menés, nous nous sommes focalisés sur les classes d'échantillonnages par réservoir dans la mesure où elles sont aisées à mettre en œuvre et garantissent un échantillon représentatif de bonne qualité. Cette approche a initialement été introduite dans [Vit85]. Le principe est le suivant : nous maintenons un réservoir de taille n, les premiers n points dans l'ensemble des données sont stockés lors de l'étape d'initialisation. Lorsque le $t^{\grave{e}me}$ point est traité, il remplace aléatoirement l'un des points du réservoir avec une probabilité de $\frac{n}{t}$. Ainsi, plus la taille de l'ensemble de données augmente plus la probabilité d'inclusion se réduit (puisque $\lim_{t\to\infty} \frac{n}{t} = 0$). Ceci est bien entendu un inconvénient majeur dans le cas des flots de données pour lesquels les informations les plus récentes dans le flot sont considérées comme les plus pertinentes [GHP+03]. Une solution pour répondre à ce problème est d'utiliser des fonctions biaisées pour réguler l'échantillon du flot de données.

L'échantillonnage par réservoir biaisé fut introduit dans [Agg06]. L'idée principale est de réguler l'introduction des points dans le réservoir et ce afin de maîtriser la *fraîcheur* de l'échantillon produit. En d'autres termes, la fonction de biais permet de moduler l'échantillon de manière à se focaliser sur des comportements récents ou plus anciens en fonction des contraintes de l'application. La fonction de biais est définie comme étant proportionnelle à la fonction $p(r,t)$ qui représente la probabilité qu'un point du flot introduit à l'instant r dans le réservoir soit encore présent à l'instant t. Aggarwal choisit dans son approche une classe spéciale de fonction de biaisage dite *exponentielle*. La fonction issue de cette classe est définie de la manière suivante : $f(r,t) = e^{\lambda(t-r)}$ où le paramètre λ correspond au taux de biais. L'originalité et la force de cette approche est que l'inclusion d'une telle fonction de biais rend possible l'utilisation d'algorithmes de remplacement simples. De plus, comme il a été prouvé dans [Agg06], la classe de fonctions de biais exponentielles implique une borne supérieure sur la taille du réservoir qui est indépendante de la longueur du flot. Pour un flot de longueur t, supposons $R(t)$ la taille maximale du réservoir qui satisfait la fonction de biais exponentielle, alors $R(t) \leqslant \frac{1}{\lambda}$.

2.2 Inégalités de concentration

Les inégalités de concentration permettent de contrôler les fluctuations de variables aléatoires. La concentration de la mesure est une question qui concerne les probabilités, l'analyse, la géométrie et la théorie de l'information. Ses aspects les plus fondamentaux se sont développés durant les 25 dernières années sous l'impulsion de Talagrand et Ledoux [LT91]. Les inégalités de concentration ont trouvé des applications en informatique dans les applications touchant à la fouille de données comme l'extraction d'itemsets [Toi96], l'échantillonnage adaptatif dynamique [SD05] et le clustering de flot [Bar02].

Ces inégalités sont nombreuses et diverses. Pour une liste exhaustive, le lecteur peut se référer à [Pap91]. Afin de faciliter la compréhension des méthodes présentées dans ce chapitre, nous introduisons les définitions de l'espérance mathématique et deux des inégalités de concentrations les plus utilisées.

$\mathbb{E}[X]$ est la moyenne arithmétique des différentes valeurs de X pondérées par leurs probabilités, sa définition formelle est :

Définition 7.1 (Espérance mathématique)**.** Soit X une variable aléatoire discrète, l'espérance est définie telle que :

$$\mathbb{E}[X] = \sum_i x_i P[X = x_i]$$

Définition 7.2 (Inégalité de Markov [Pap91])**.** Soit X une variable aléatoire non-négative alors pour tout $t > 0$ on a :

$$P[X \geqslant t] \leqslant \frac{\mathbb{E}[X]}{t} \tag{7.1}$$

Démonstration. Pour un événement quelconque E, soit $\mathbb{I}_E$ la variable aléatoire indicatrice de E (i.e $\mathbb{I}_E = 1$ si E apparaît et $\mathbb{I}_E = 0$ sinon).

On a alors $\mathbb{I}_{[X \geqslant t]} = 1$ si l'événement $[X \geqslant t]$ se produit et $\mathbb{I}_{[X \geqslant t]} = 0$ si $X < t$. Alors pour $t > 0$: $t.\mathbb{I}_{[X \geqslant t]} \leqslant X$ et donc $\mathbb{E}[t\mathbb{I}_{[X \geqslant t]}] \leqslant \mathbb{E}[X]$. En remarquant que $t.\mathbb{E}[\mathbb{I}_{[X \geqslant t]}] = t.P[X \geqslant t]$, on a :

$$t.P[X \geqslant t] \leqslant \mathbb{E}[X]$$

Et comme $t > 0$:

$$P[X \geqslant t] \leqslant \frac{\mathbb{E}[X]}{t}$$

□

Exemple 7.1. Supposons que les manuscrits de thèse dans le domaine de la fouille de données font en moyenne 100 pages. Quelle est la proportion de thèses ayant plus de 1000 pages ?

Supposons X la variable aléatoire discrète associée au *"nombre de pages du manuscrit choisi"*, ici $\mathbb{E}[X] = 100$, alors par l'inégalité de Markov on a :

$$P[X \geqslant 1000] \leqslant \frac{\mathbb{E}[X]}{1000} = \frac{1}{10}$$

Définition 7.3 (Inégalité de Hoeffding [Hoe63])**.** Soit $X_1, X_2, \ldots, X_n$ des variables aléatoires indépendantes telles que $X_i \in [a, b]$ avec une probabilité de 1. Soit S la somme de ces variables alors pour tout $t > 0$:

$$\begin{aligned} P[S - \mathbb{E}[S] \geqslant nt] &\leqslant exp\left(\frac{-2nt^2}{(b-a)^2}\right) \\ P[\ |S - \mathbb{E}[S]\ | \geqslant nt] &\leqslant 2exp\left(\frac{-2nt^2}{(b-a)^2}\right) \end{aligned} \tag{7.2}$$

3 Échantillonnage sur les bases de données statiques

Dans cette section nous discutons de l'utilité de l'échantillonnage en tant qu'étape de prétraitement dans l'extraction de motifs séquentiels dans le cadre des bases de données statiques. Étant donné que l'un des facteurs clés pour l'extraction est la taille de la base considérée, l'intuition sous-jacente est que n'importe quel algorithme d'extraction pourrait être lancé sur un échantillon de la base de données originale afin d'avoir des résultats de manière plus rapide et plus facile.

Exemple 7.2. Supposons que $\mathcal{D}$ soit une base de données d'une compagnie de téléphonies et télécommunications contenant l'historique des appels de 20 millions de clients sous forme de séquences. L'extraction de motifs séquentiels sur toute la base de données va générer une grande activité en terme d'entrées-sorties et les données ne seront probablement pas chargées en mémoire. Il est donc plus utile de générer un échantillon de cette base.

La première question qui se pose si nous voulons extraire des motifs à partir d'un échantillon est : *à quel point l'échantillon est-il pertinent par rapport au jeu de données original ?*

Nous y répondons en exhibant une garantie sur le taux d'erreur du support d'une séquence. Notons qu'une approche similaire a été proposée pour l'extraction d'itemsets fréquents dans [Toi96] dans le cadre d'un algorithme d'extraction d'itemsets fréquents *sans approximations.*

Définition 7.4 (Taux d'erreur)**.** Soit $\mathcal{D}$ une base de données transactionnelle de clients et notons $\mathcal{S}_\mathcal{D}$ l'échantillon aléatoire généré à partir de $\mathcal{D}$. Soit s une séquence présente dans $\mathcal{D}$. Le taux d'erreur absolu en terme d'estimation du support, noté $e(s, \mathcal{S}_\mathcal{D})$, est défini par :

$$e(s, \mathcal{S}_\mathcal{D}) = |Support(s, \mathcal{S}_\mathcal{D}) - Support(s, \mathcal{D})|$$

Soit $X_{i,s}$ une variable aléatoire indépendante définie telle que :

$$\begin{cases} Pr[X_{i,s} = 1] = p_i & \text{si le } i^{eme} \text{ client supporte la séquence } s, \\ Pr[X_{i,s} = 0] = 1 - p_i & \text{sinon.} \end{cases} \tag{7.3}$$

notons $X(s, \mathcal{S}_\mathcal{D}) = \sum_i^{|\mathcal{S}_\mathcal{D}|} X_{i,s}$.

$X(s, \mathcal{S}_\mathcal{D})$ représente le nombre de séquences dans l'échantillon $\mathcal{S}_\mathcal{D}$ qui supportent la séquence s. Cette quantité peut être réécrite de la manière suivante :

$$X(s, \mathcal{S}_\mathcal{D}) = Support(s, \mathcal{S}_\mathcal{D}).|\mathcal{S}_\mathcal{D}| \tag{7.4}$$

De même, l'espérance de la variable $X(s, \mathcal{S}_\mathcal{D})$ est $\mathbb{E}[X(s, \mathcal{S}_\mathcal{D})] = Support(s, \mathcal{D}).|\mathcal{S}_\mathcal{D}|$.

Nous voulons estimer la probabilité que le taux d'erreur $e(s, \mathcal{S}_\mathcal{D})$ lors d'un échantillonnage dépasse un certain seuil d'erreur ε défini par l'utilisateur,(i.e $Pr[e(s, \mathcal{S}_\mathcal{D}) > \varepsilon]$). Cette estimation est généralement appelé dans la littérature une (ε, δ)-approximation.

Pour obtenir l'estimation, nous utilisons l'inégalité de Hoeffding [Hoe63]. Le théorème suivant exhibe une borne inférieure sur la taille de l'échantillon.

Théorème 7.1. Soit s une séquence et $|\mathcal{S}_\mathcal{D}|$ la taille de l'échantillon, alors nous avons :

$$Pr[e(s, \mathcal{S}_\mathcal{D}) > \varepsilon] \leqslant \delta \; si \; |\mathcal{S}_\mathcal{D}| \geqslant \ln(\frac{2}{\delta})\frac{1}{2\varepsilon^2} \tag{7.5}$$

Démonstration.

$$\begin{aligned} Pr[e(s, \mathcal{S}_\mathcal{D}) > \varepsilon] &= Pr[\ |Support(s, \mathcal{S}_\mathcal{D}) - Support(s, \mathcal{D})| > \varepsilon] \\ &= Pr[\ |Support(s, \mathcal{S}_\mathcal{D}).|\mathcal{S}_\mathcal{D}| - Support(s, \mathcal{D}).|\mathcal{S}_\mathcal{D}| \ | > \varepsilon.|\mathcal{S}_\mathcal{D}|] \\ &= Pr[\ |X(s, \mathcal{S}_\mathcal{D}) - E[X(s, \mathcal{S}_\mathcal{D})] \ | \ > \varepsilon.|\mathcal{S}_\mathcal{D}|] \end{aligned} \tag{7.6}$$

Par l'inégalité de Hoeffding (Définition 7.3) nous avons :

$$Pr[\ |X(s, \mathcal{S}_\mathcal{D}) - \mathbb{E}[X(s, \mathcal{S}_\mathcal{D})] \ | \ > \varepsilon.|\mathcal{S}_\mathcal{D}| \ ] \leqslant \delta \tag{7.7}$$

avec $\delta = 2e^{-2\varepsilon^2.|\mathcal{S}_\mathcal{D}|}$ □

Il est à noter que les inégalités de Hoeffding sont souvent considérées comme plus générales et moins précises que l'inégalité de concentration de Chernoff [Che52], mais le choix de variables aléatoires indépendantes dans l'intervalle $[0, 1]$ donne des résultats similaires pour ces deux inégalités. De plus, il est important de remarquer que la taille de l'échantillon $\mathcal{S}_\mathcal{D}$ est conditionnée uniquement par ε et δ.

La table 7.1 illustre quelques exemples de tailles d'échantillons en terme de séquences pour différentes valeurs de ε et δ. Nous pouvons remarquer que lorsque les valeurs ε et δ sont trop strictes, l'échantillon peut atteindre une taille assez importante.

ε	δ	$\lvert\mathcal{S}_\mathcal{D}\rvert$
0.01	0.01	26492
0.01	0.001	38005
0.01	0.0001	49518
0.001	0.27	1000000
0.001	0.01	2649160
0.001	0.0025	3333333

Tab. 7.1: Différentes tailles d'échantillons pour ε et δ donnés

Nous allons voir maintenant comment peut-on étendre ce résultat afin de l'incorporer dans un contexte d'extraction de motifs séquentiels dans le modèle de flot de données.

4 Motifs séquentiels, échantillonnage et flots de données

Un des problèmes majeurs qui rend la pratique de l'échantillonnage difficile sur les flots est que l'on ne sait pas à l'avance la taille du flot. Il faut donc développer des algorithmes d'échantillonnages dynamiques qui prennent en compte l'évolution et les changements dans la distribution des données transitant sur le flot [Agg07].

Dans cette section, nous étendons les résultats précédents sur l'échantillonnage dans le cadre des bases statiques. Nous présentons un algorithme de maintien dynamique d'échantillon (biaisé ou non) et qui prend en compte les différentes évolutions du flot de données tout en respectant certaines contraintes sur l'ordre d'échantillonnage des itemsets présents dans les différentes séquences. L'algorithme présenté peut être vu comme une étape de pré-traitement nécessaire afin de permettre l'extraction de séquences fréquentes. Afin que cette étape de pré-traitement soit pertinente, elle doit respecter les conditions suivantes :

1. L'échantillon doit avoir une borne inférieure sur sa taille afin de minimiser le taux d'erreur absolu en terme d'estimation du support.
2. A cause de la nature même des séquences, les opérations d'insertions et d'enlèvements, nécessaires pour la mise à jour d'un échantillon, doivent se faire au niveau des identifiants des séquences, mais aussi de leurs itemsets. Pour s'en convaincre, il suffit de considérer un ensemble fini de séquences avec pour chacun d'eux un grand nombre d'itemsets qui se rajoutent à chaque instant t. Cet ensemble ne peut bien sûr pas être considéré comme un échantillon ou un réservoir car il n'est pas borné et ne fait qu'augmenter avec le flot.

Dans notre modèle de flot de données, un point de données apparaissant à chaque instant t est défini comme un couple constitué d'un identifiant de séquence et d'un itemset. Ce modèle est très efficace pour représenter les flots issus des réseaux de télécommunications comme par exemple un réseau TCP/IP où un point du flot consisterait en un paquet TCP/IP contenant comme identifiant l'adresse IP source et où les itemsets correspondraient aux différents attributs du paquet TCP/IP.

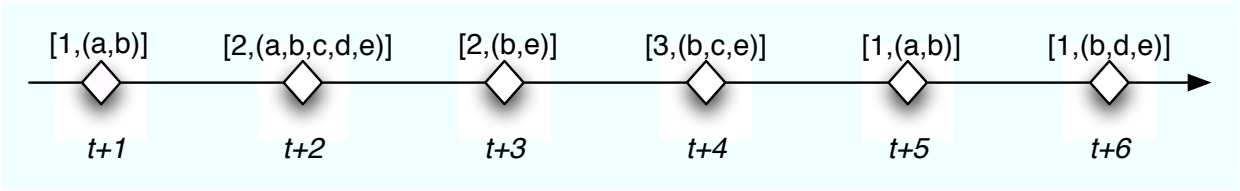


Fig. 7.1: Modèle de flot contenant 6 points avec 3 identifiants de séquences

Ce modèle peut être aussi vu comme un tableau contenant dans ses lignes les identifiants de séquences et dans ses colonnes les différents itemsets apparaissant sur le flot. La table 7.2 illustre une représentation tabulaire de l'exemple du flot de la figure 7.1.

Id. Séquence	Itemsets
1	$(a,b)(a,b)(b,d,e)$
2	$(a,b,c,d,e)(b,e)$
3	(b,c,e)

Tab. 7.2: Modèle de flot représentant les points arrivés sous forme de tableau

Partant de ces contraintes et des résultats théoriques obtenus dans la section 3, nous proposons un algorithme de remplacement issu de l'approche de réservoir biaisé proposée dans [Agg06] qui permet de réguler l'échantillonnage des itemsets des séquences sur le flot grâce à une fonction de biaisage temporelle exponentielle.

Le principe général est le suivant : nous commençons avec un réservoir vide, de capacité maximale $\frac{1}{\lambda}$ (nous discutons un peu plus tard la valeur du taux de biais λ) et chaque itemset d'une séquence apparaissant sur le flot est inséré de manière probabiliste dans le réservoir après une opération de *lancer de pièce* : soit par un remplacement des itemsets d'une séquence déjà présente dans le réservoir, soit par un ajout direct dans une des places encore vacantes. Comme nous l'avons vu précédemment, nous devons aussi bien contrôler la taille du réservoir en terme de nombre de séquences qu'en nombre d'itemsets dans chacune de ces séquences. Cette opération de contrôle est appliquée grâce à une approche de *fenêtre glissante* qui permet de garder uniquement les itemsets les plus récents pour une séquence donnée dans le réservoir. Une fenêtre glissante peut être définie soit comme une fenêtre basée sur les séquences de taille k, contenant les k itemsets des points les plus récents apparus sur le flot, soit comme une fenêtre basée sur un intervalle de temps de taille t contenant tous les points apparus sur le flot sur une durée de temps t.

Dans notre approche nous utilisons des fenêtres glissantes basées sur des séquences afin de garder uniquement les transactions les plus récentes pour les clients présents dans l'échantillon. Ce type de fenêtre glissante permet l'extraction de séquences sur un horizon récent du flot. De plus, la fonction exponentielle de biais permet à l'utilisateur de choisir la taille de son réservoir (avec des contraintes sur l'(ε,δ)-approximation) et ainsi, un échantillon représentatif du flot peut être construit et mis à jour en mémoire selon les besoins de l'application et de l'utilisateur. Le corollaire suivant, issu du théorème 7.1 exhibe le lien qui existe entre le taux de biais λ et les seuils d'erreurs ε et δ :

Corollaire 7.1. Soient λ le taux de biais, ε le seuil d'erreur et δ la probabilité maximale telle que $e(s,\mathcal{S}_{\mathcal{D}}) > \varepsilon$, alors :

$$\lambda \leqslant \frac{2\varepsilon^2}{ln(2/\delta)} \tag{7.8}$$

Démonstration. Par [Agg06], pour un flot de taille t, supposons $R(t)$ la taille maximale possible du réservoir qui satisfait la fonction de biaisage exponentielle, on a alors par définition : $R(t) \leqslant \frac{1}{\lambda}$. Nous considérons le réservoir entier comme l'échantillon cible pour l'extraction de motifs, on a donc : $R(t) = |\mathcal{S}_{\mathcal{D}}|$. Par substitution dans le théorème 7.1, nous obtenons le résultat énoncé. □

Exemple 7.1. Supposons qu'un utilisateur définisse le seuil d'erreur $\varepsilon = 0.01$ et $\delta = 0.01$ pour les besoins de son application d'extraction de motifs, alors pour avoir des résultats pertinents la condition $\lambda \leqslant 0.000377$ doit être respectée. Ceci veut dire que l'utilisateur doit choisir un taux de biaisage $\lambda \in [0, 0.000377]$ s'il veut être assuré que les résultats de son extraction de motifs séquentiels respectent la (ε, δ)-approximation qu'il a choisi. ainsi, dans ce cas là, la taille du réservoir sera au minimum de 26492 séquences.

La table 7.3 montre quelques valeurs du taux de biaisage λ et la taille minimale du réservoir nécessaire pour la bonne approximation du support des séquences.

ε	δ	λ	$\min(R(t))$
0.01	0.01	0.0000377	26492
0.01	0.001	0.00002631	38005
0.01	0.0001	0.00002019	49518
0.001	0.27	0.000001	1000000
0.001	0.0025	0.0000003	3333333

Tab. 7.3: Différents taux de biaisage λ pour différentes valeurs de ε et δ

4.1 Algorithme

Le fonctionnement général de l'algorithme est le suivant :

1. A l'arrivée d'un itemset d'une séquence S_i, voir si la séquence est dans la liste noire (nous discutons la nécessité de cette liste noire un peu plus loin), auquel cas ignorer ce point, sinon passer à l'étape 2.
2. Voir si la séquence S_i est déjà dans le réservoir, si oui, rajouter l'itemset dans la fenêtre et voir s'il y a un décalage nécessaire à faire sinon passer à l'étape 3.
3. Faire un *lancer de pièce* (tir aléatoire), en cas de succès remplacer une des séquences déjà présente dans le réservoir de manière aléatoire (la séquence remplacée est alors mise dans la

liste noire). En cas d'échec, introduire la séquence et son itemset dans le réservoir sans rien remplacer.

La partie la plus importante dans l'algorithme est la gestion des décalages des fenêtres glissantes et la mise à jour de la liste noire. Nous détaillons maintenant ces étapes.

Un problème peut apparaître lors de l'étape de remplacement des clients présents dans le réservoir : nous devons détecter si une séquence était déjà présente dans le réservoir, car la réintroduire sans aucun test préalable rendra les résultats de l'extraction des motifs séquentiels inconsistants avec la réalité du flot. Le problème d'inconsistance apparaît lorsqu'une séquence est remplacée par une autre et qu'un de ses itemsets associés revient dans le réservoir à un instant ultérieur. Au moment de l'extraction, cette séquence n'aura pas tous les itemsets qu'elle aurait dû avoir dans la fenêtre glissante actuelle(nous discutons ce point dans l'exemple suivant). Les points de certaines séquences doivent donc être ignorés. Mais d'un autre côté, ignorer des points sur le flot peut introduire un nouveau biais, puisque seules les séquences ayant remplacées d'autres séquences seront présentes dans l'échantillon. Afin de résoudre ce problème d'inconsistance entre les itemsets des différentes séquences nous introduisons un système de *liste noire* qui permet d'interdire l'échantillonnage à un certain nombre de séquences indésirables. Cette liste noire n'est pas irréversible et est réactualisée à chaque glissement de la fenêtre. L'exemple 7.3 illustre un cas d'inconsistance et sa gestion avec la liste noire.

Exemple 7.3. Soit $\mathcal{R}$ le réservoir de taille maximale 2 et $W = 4$, la taille de la fenêtre glissante pour chaque séquence dans le réservoir (i.e. Il peut y avoir au plus 2 séquences introduites dans le réservoir et chacune de ces séquences peut avoir jusqu'à 4 itemsets). Supposons pour les besoins de l'exemple qu'à un instant t, le réservoir est rempli et contient les itemsets des séquences S_1 et S_2 (Figure 7.2). Les prochains points du flots sont schématisés dans la figure 7.3.

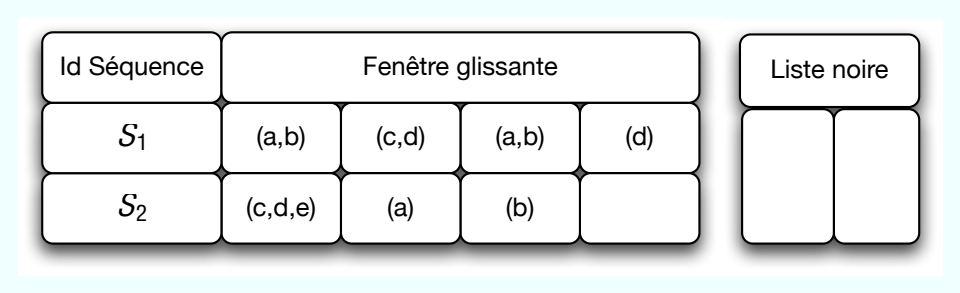


Fig. 7.2: Réservoir et liste noire à l'instant t

– A l'instant $t+1$ le point apparaissant sur le flot est constitué par l'identifiant de séquence S_3 et l'itemset (d, e) et on suppose qu'il remplace la séquence S_2 dans le réservoir après un lancer réussi. Nous adoptons ici la même approche que dans [Agg06], c'est-à-dire que le lancer est

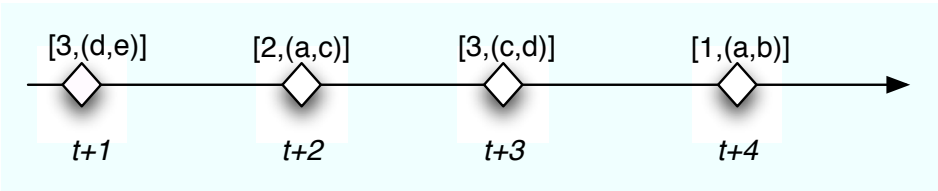


FIG. 7.3: Points du flot utilisés dans l'exemple 7.3

considéré réussi si la valeur $coin \in [0, 1]$ tirée aléatoirement est plus petite que le taux de remplissage du réservoir à l'instant $t + 1$, noté $F(t + 1)$. Dans notre exemple, la capacité maximale du réservoir est de 2 et à l'instant $t + 1$, $\mathcal{R}$ contient 2 séquences introduites, d'où $F(t + 1) = \frac{2}{2} = 1$. La liste noire est mise à jour et une nouvelle entrée est enregistrée pour la séquence S_2 avec 3 glissements de fenêtres.

Id Séquence	Fenêtre glissante				Liste noire	
S_1	(a,b)	(c,d)	(a,b)	(d)	S_2	3
S_3	(d,e)					

FIG. 7.4: Réservoir et liste noire à l'instant $t + 1$

– A l'instant $t + 2$, le point apparaissant sur le flot provient de la séquence S_2. Il est clair que pour des raisons de consistance, notre algorithme ne peut pas réintroduire dans le réservoir le prochain point de la séquence S_2 (le point ayant pour itemset (a, c)). Ainsi, si un algorithme d'extraction de motifs séquentiels était appliqué directement après l'instant $t+2$, les résultats seraient erronés, car il ne prendrait pas en compte la sous-séquence d'itemsets déjà ignorée et effacée de S_2 à l'instant $t + 1 : (c, d, e)(a)(b)$. Il faut donc attendre pour les points de la séquence S_2, 3 glissements de fenêtres (la taille de la sous-séquence effacée à l'instant $t + 1$), avant de pouvoir être réintroduits dans le réservoir.
– A l'instant $t + 3$, le point est directement rajouté au réservoir car la séquence S_3 est déjà présente dans le réservoir, on rajoute simplement l'itemset (c, d) dans la fenêtre glissante de S_3.
– A l'instant $t + 4$, le point de donnée est rajouté à la fenêtre glissante de la séquence S_1, ce dernier ajout va forcer un glissement de fenêtre pour toutes les séquences présentes dans le réservoir puisque la taille maximale de la fenêtre est définie à 4 itemsets au maximum. La liste noire est mise à jour afin de prendre en compte ce glissement et toutes les valeurs

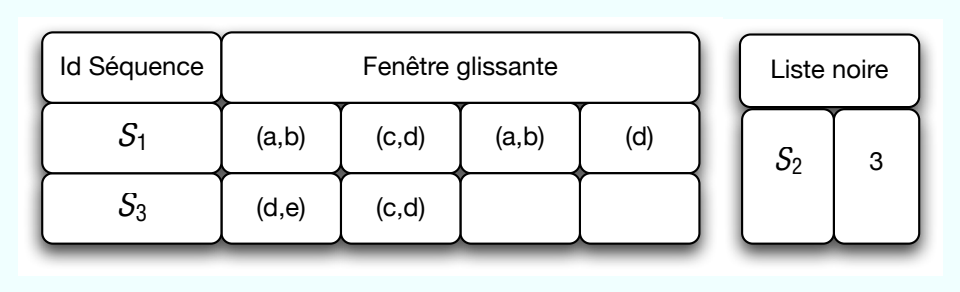


FIG. 7.5: Réservoir et liste noire à l'instant $t + 3$

sont décrémentées de 1. Les points de la séquence S_2 ne pourront être réintroduits dans le réservoir que lorsque la valeur du compteur devient égale à 0 dans la liste noire.

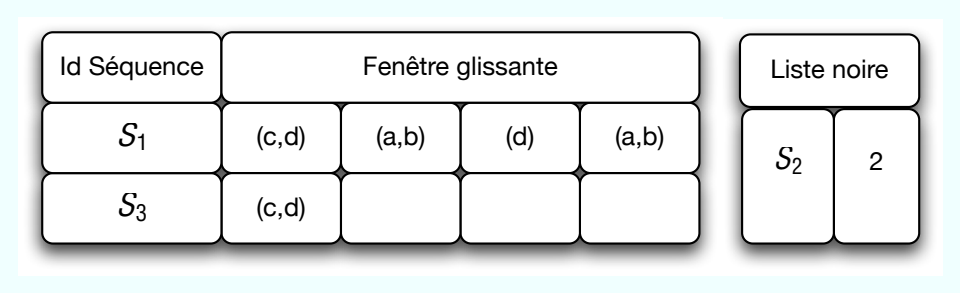


FIG. 7.6: Réservoir et liste noire à l'instant $t + 4$

L'algorithme 5 détaille l'exécution complète de notre algorithme.

Algorithme 5 : Algorithme d'échantillonnage par réservoir biaisé pour l'extraction de motifs séquentiels

Data : Réservoir $\mathcal{S}_\mathcal{D}$;Taille du réservoir : n ; Nombre de séquences dans le réservoir : q ; Taille de la fenêtre glissante : $|W|$; point P.

Result : Réservoir mis à jour après le traitement du point P.

```
1  begin
2      // BL : la liste noire
3      if P.S_i ∉ BL then
4          if P.S_i ∉ S_D then
5              // Insertion avec lancer
6              Coin ← Random(0, 1);
7              if Coin ⩽ q/n then
8                  pos ← Random(0, q);
9                  Replace(S_pos, P.S_i);
10                 BL.add(P.S_pos);
11             else
12                 Add(T, S_D) ;q + +;
13         else
14             //Insertion directe
15             Insert(S_i,T);
16             if S_i.window.size > |W| then
17                 slideAllWindows();
18                 Update(BL);
19 end
```

Tout au long de la section 4, nous avons supposé implicitement que l'algorithme 5 construisait un réservoir biaisé respectant la fonction de biaisage exponentielle avec le paramètre λ. Nous allons maintenant présenter une preuve formelle de cette proposition. Comme dans [Agg06], nous allons démontrer que la politique de remplacement appliquée dans notre algorithme permet de construire un réservoir biaisé de taille $|\mathcal{S}_\mathcal{D}| = n$ avec $\lambda = \frac{1}{n}$.

Proposition 7.1. L'algorithme 5 construit un réservoir biaisé respectant la fonction de biaisage temporel $f(r,t) = e^{-\lambda(t-r)}$ avec $\lambda = \frac{1}{n}$.

Démonstration. Soient K le nombre total de clients, q_t le nombre de points déjà insérés dans le réservoir à l'instant t et b_t la taille de la liste noire à l'instant t. Posons $n = |\mathcal{S}_\mathcal{D}|$, le nombre possible de séquences dans le réservoir.

A un instant i, la probabilité que le client arrivant sur le flot soit dans la liste noire est $\frac{b_i}{K}$ et la probabilité que le client soit déjà dans le réservoir est $\frac{q_i}{K}$. De plus, la probabilité que le lancer de pièce soit un succès après le test d'appartenance à la liste noire est $\frac{q_i}{n}$, la probabilité dans le cas d'un lancer fructueux est $\frac{1}{q_i}$ donc la probabilité qu'un point dans le réservoir soit éjecté à l'instant donné i est $\frac{q_i}{n} \times \frac{1}{q_i} = \frac{1}{n}$ (la probabilité qu'un point soit encore à l'instant donné i dans le réservoir est alors $(1 - \frac{1}{n})$, étape 2 de l'algorithme). De plus, sachant que $r < t$ nous devons calculer toutes les combinaisons possibles (r, t) avec t fixé de la probabilité qu'un client inséré à l'instant r soit encore dans le réservoir à l'instant t :

$$\begin{aligned}\prod_{i=r}^{t}\left(\frac{b_i}{K} + (1-\frac{b_i}{K})\left[\frac{q_i}{K} + (1-\frac{q_i}{K})(1-\frac{1}{n})\right]\right) &= \prod_{i=r}^{t}\left(1 + \frac{b_i+q_i}{K.n} - \frac{b_i.q_i}{K^2.n} - \frac{1}{n}\right) \\ &= \prod_{i=r}^{t}\left(1 + \frac{q_i}{K.n} + \frac{b_i.(K-q_i)}{K^2.n} - \frac{1}{n}\right)\end{aligned} \tag{7.9}$$

Pour de très grandes valeurs de K, $\frac{q_i}{K.n} \simeq K - q_i \simeq K$, donc $\frac{b_i.(K-q_i)}{K^2.n} \simeq 0$. D'où,

$$\simeq \prod_{i=r}^{t}\left(1 - \frac{1}{n}\right) = \left[\left(1 - \frac{1}{n}\right)^n\right]^{(t-r)/n} \tag{7.10}$$

Pour de grandes valeurs de n, $\left(1 - \frac{1}{n}\right)$ est approximativement égal à e^{-1}. Par substitution dans l'équation 7.10, nous avons le résultat énoncé. □

5 Expérimentations

$\|D\|$	Nombre de séquences de données
$\|T\|$	Nombre moyen de transactions par séquences de données
$\|N\|$	Nombre moyen d'items par transaction
$\|I\|$	Nombre d'items dans la base
$\|S\|$	Taille de la base en Go

Tab. 7.4: Paramètres utilisés dans les différents jeux de données

Pour évaluer les performances de nos propositions d'échantillonnage nous avons réalisé des expériences sur des jeux de données synthétiques et réels. Le but étant est de répondre aux questions suivantes :

1. *Est-ce que le processus d'extraction reste précis lorsque l'on travaille sur un échantillon du jeu de données original ?*
2. *Le processus d'extraction sur les flots reste-t-il borné en terme d'utilisation mémoire si nous appliquons notre méthode de pré-traitement ?*

Toutes les expérimentations ont été lancées sur un ordinateur MacBookPro Core-Duo 2.16 Ghz doté de 1Go de mémoire, tournant sous Mac OS X 10.5.2. Tous les algorithmes ont été codés dans le langage de programmation C++. Pour comparer nos résultats, nous utilisons une implémentation personnelle de PrefixSpan[1] [PHMa+01], qui permet l'extraction des motifs séquentiels. Les tests lancés sur les jeux de données synthétiques sont générés par le logiciel QUEST[2]. Les différents jeux de données utilisés dans ces expérimentations et leurs caractéristiques sont regroupés dans le tableau 7.5. Afin d'échantillonner les bases de données statiques nous avons implémenté l'algorithme d'échantillonnage par réservoir de [Vit85].

Data Set	$\|I\|$	$\|N\|$	$\|T\|$	$\|D\|$	$\|S\|$	Utilisation
CL1MTR2.5SL50IT10K	10000	2.5	50	1000000	1.05	flot
CL1MTR10SL20IT10K	10000	10	20	1000000	0.797	statique
CL0.5MTR20SL20IT10K	10000	20	20	500000	0.767	statique
CL6MTR2.5SL10IT20K	20000	2.5	10	6000000	0.686	statique

Tab. 7.5: Différents jeux de données synthétiques utilisés pour les experimentations.

5.1 Expérimentations sur les bases de données statiques

Nous avons testé la pertinence de nos calculs théoriques pour l'échantillonnage des bases de données statiques (section 3). Nous avons utilisé dans nos expérimentations les trois derniers jeux de données présentés dans le tableau 7.5 afin de comparer les résultats classiques (voir Figure 7.5) aux estimations de l'(ε, δ)-approximation du théorème 7.1. Les résultats des différentes expérimentations sont listés dans les tableaux 7.6 et 7.7. Nous avons construit des échantillons de différentes tailles allant de 25000 à 500000 séquences en utilisant l'approche d'échantillonnage par réservoir. Les expérimentations ont été répétées 5 fois pour chaque échantillon et les valeurs présentées sont les moyennes des résultats. La colonne *Erreur* dans les tableaux décrit le nombre de séquences extraites dont le support dépasse nos (ε, δ)-approximations ainsi que les faux-positifs (les séquences fréquentes dans l'échantillon alors qu'elles ne le sont pas dans la base de données)

[1] la version de PrefixSpan situé sur le site web http ://illimine.cs.uiuc.edu/ étant uniquement une version binaire pour Linux et Windows.

[2] http ://www.almaden.ibm.com/cs/projects/iis/hdb/Projects/data_mining/datasets/syndata.html

et les vrais-négatifs (les séquences non fréquentes dans l'échantillon alors qu'elle les sont dans la base de données). On remarque que l'échantillonnage sur les bases de données statiques reste très précis, comme cela a été présenté dans les résultats théoriques de la section 3 et ce même avec de très petites tailles de résumés. Le temps d'extraction de motifs et l'utilisation mémoire pour ce processus est diminué de plusieurs ordres de magnitudes ce qui permet de pousser le processus global d'extraction vers des supports très bas.

Nom	# de motifs seq. extraits	Support	Traitement	Mémoire ut.
CL1MTR10SL20IT10K	715	7%	537.165 s.	578.587 Mo
CL0.5MTR20SL20IT10K	582	15%	538.560 s.	502.034 Mo
CL6MTR2.5SL10IT20K	5503	0.3%	523.969 s.	685.821 Mo

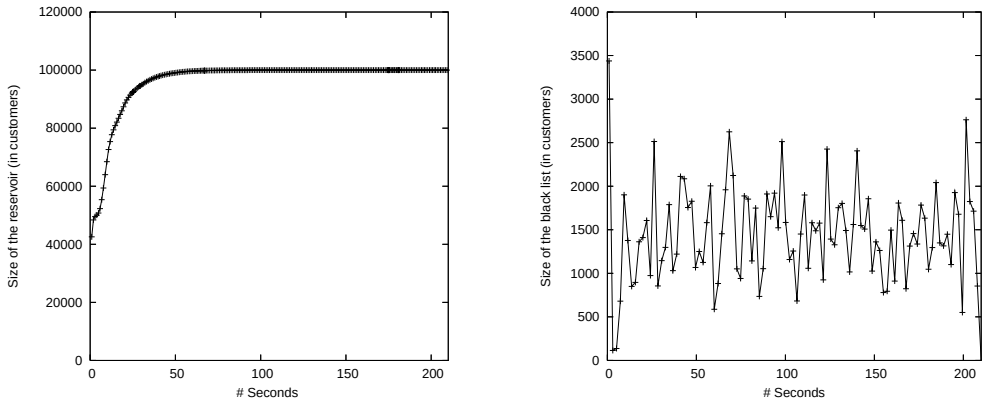
Tab. 7.6: Résultats de l'extraction de motifs sur les jeux de données statiques

5.2 Expérimentations sur les flots de données

L'expérimentation sur les flots de données utilise le jeu de données CL1MTR2.5SL50IT10K en faisant varier différents paramètres : la valeurs de biais (λ) et la taille de la fenêtre glissante. Dans ces expérimentations nous mettons en valeur l'efficacité de l'échantillonnage et montrons empiriquement que la liste noire reste bornée dans le temps. Les figures 7.7 et 7.8 montrent que la liste noire, qui est la garante de la consistance de l'échantillon pour le processus d'extraction de motifs reste assez limitée en terme d'espace mémoire utilisé et ce grâce aux différentes mises à jours faites à chaque décalage des fenêtres glissantes des séquences présentes dans le réservoir. De plus, si la taille de la fenêtre glissante est petite, la liste noire tend à être trés petite aussi, ceci est due aux fréquents décalages qui permettent de mettre à jour les clients ignorés dans le réservoir. Les figures 7.7 et 7.8 soulignent aussi le fait que le réservoir se remplit très vite, l'étape d'extraction de motifs séquentiels peut donc être lancé à partir de la $50^{\text{ème}}$ seconde dans la première expérimentation, et à partir de la $400^{\text{ème}}$. Pour ce qui est du besoin en espace mémoire, nous pouvons voir sur la figure 7.9 que les réservoirs lors des différentes expérimentations restent bornées et stables en terme d'occupation mémoire.

CL0.5MTR20SL20IT10K					
$\|\mathcal{S}_\mathcal{D}\|$	# de motifs. séq.	Erreurs	Temps requis	Mémoire requise	Taille de l'échantillon
25000	582	2	10.341 s.	25.128 MB	38 MB
38005	579	3	13.640 s.	38.18 MB	57.9 MB
50000	582	1	20.367 s.	50.419 MB	76.2 MB
100000	580	3	44.410 s.	100.001 MB	152.1 MB
200000	582	1	158.566 s.	199.823 MB	205.4 MB
CL1MTR10SL20IT10K					
$\|\mathcal{S}_\mathcal{D}\|$	# de motifs. séq.	Erreurs	Temps requis	Mémoire requise	Taille de l'échantillon
25000	704	14	4.878 s.	14.291 MB	19.9 Mo
38005	707	9	5.899 s.	21.752 MB	30.2 Mo
50000	716	3	7.592 s.	28.982 MB	39.9 Mo
100000	717	2	15.339 s.	57.984 MB	79.8 Mo
200000	715	1	36.265 s.	115.579 MB	159.2 Mo
CL6MTR2.5SL10IT20K					
$\|\mathcal{S}_\mathcal{D}\|$	# de motifs. séq.	Erreurs	Temps requis	Mémoire requise	Taille de l'échantillon
25000	5830	342	1.6 s.	3.070 MB	3 Mo
38005	5239	271	1.608 s.	4.503 MB	4.4 Mo
50000	5321	184	1.903 s.	5.997 MB	5.9 Mo
100000	5432	75	2.725 s.	11.831 MB	11.8 Mo
500000	5531	31	9.854 s.	58.947 MB	58.8 Mo

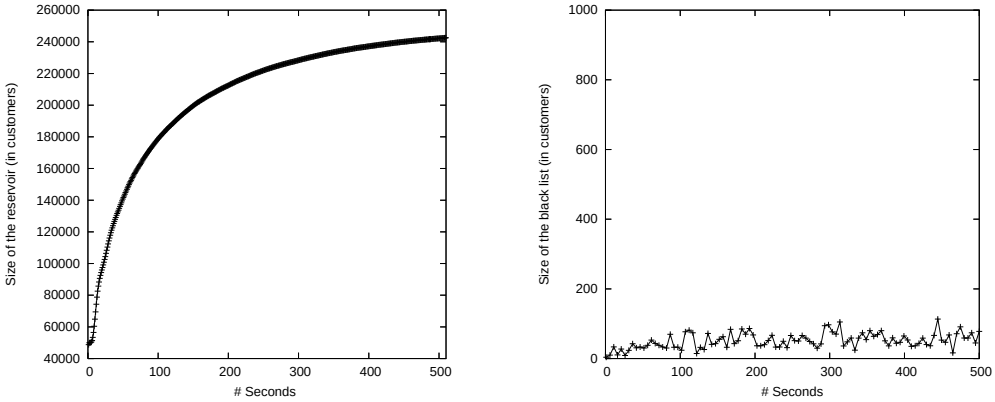
Tab. 7.7: Résultats d'extractions pour les différents jeux de données statiques



(a) Taille du réservoir en terme de clients

(b) Taille de la liste noire en terme de clients

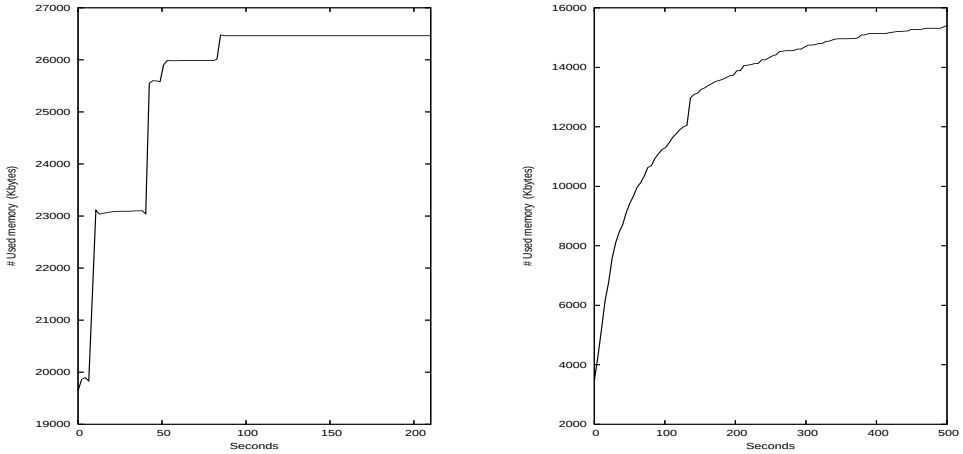
Fig. 7.7: Expérimentations avec $\lambda = 0.00001$et $|W| = 10$



(a) Taille du réservoir en terme de clients

(b) Taille de la liste noire en terme de clients

Fig. 7.8: Expérimentations avec $\lambda = 0.00004$et $|W| = 2$.



(a) λ=0.00001 et $|W| = 10$

(b) λ=0.00004 et $|W| = 2$

Fig. 7.9: Besoin mémoire pour le jeu de données CL1MTR2.5SL50IT10K

6 Discussion

Dans ce chapitre, nous nous sommes intéressés à de nouvelles techniques de résumés pour représenter des bases de données de motifs séquentiels. Nous avons montré qu'une approche basée sur des échantillons était tout à fait adaptée pour des bases de données statiques et que nous étions capables de maîtriser les taux d'erreur dans les résultats d'extraction de motifs séquentiels. Dans le cadre plus spécifique de l'extraction de motifs séquentiels sur les flots de données, il existe quelques propositions comme SPEED [RPT06] et MDMS [MM06], mais il n'y a à notre connaissance aucune méthode d'extraction de motifs séquentiels basé sur une étape de pré-traitement consistant à échantillonner le flot. La principale raison est la complexité d'une telle mise en oeuvre, puisque dans ce cas là, la difficulté majeure est de prendre en compte la structure sous-jacente des motifs dans le processus d'échantillonnage[1]. Dans le cadre des motifs séquentiels, cela revient à respecter et intégrer dans l'échantillonnage l'ordre d'apparition des itemsets dans les différentes séquences formant le flot.

Dans cette discussion, nous revenons à présent sur les résultats approximés obtenus. Bien entendu, en proposant une approche de ce type, nous avons montré que pour l'utilisation de bases de données statiques, nous étions capables d'offrir une borne sur l'échantillon afin que celui-ci soit suffisamment représentatif de la base de données d'origine. Même si cela ne rentre pas dans nos objectifs il est important de préciser que cette approche peut également facilement être étendue, pour les bases de données statiques, afin d'extraire en deux passes sur la base d'origine l'ensemble des motifs séquentiels fréquents et offrir ainsi une réponse complète et non plus approximative. En effet, en appliquant le principe de l'algorithme SAMPLING proposé par Toivonen [Toi96], il nous suffit d'extraire sur l'échantillon obtenu d'une part les séquences fréquentes mais également les éléments de la bordure négative. Ensuite, lors d'un parcours sur la base de données complète, nous examinons les fréquences des séquences et des éléments de la bordure. Enfin, le second parcours sur la base garantit de ne chercher que les éléments de la bordure qui n'avaient pas été traités lors de la première passe. Nous pouvons noter qu'une approche similaire de gestion de la bordure négative a été proposée par Parthasarathy et al. [PZOD99] dans le cas de bases de données incrémentales.

Les multiples perspectives associées à ce travail telles que l'utilisation de techniques de martingales afin de borner théoriquement la taille de la liste noire ou les algorithmes d'échantillonnages *online* sont discutés dans le dernier chapitre de ce manuscrit.

[1] Cette remarque s'applique d'ailleurs aux arbres, graphes et tous les autres motifs structurés

Chapitre 8

Extraction de motifs multidimensionnels sur les flots

Nous avons vu précédemment que la prise en compte des contraintes associées aux flots permettait d'adapter des algorithmes d'extraction de motifs séquentiels en se basant sur des techniques d'échantillonnages. Toutefois, les algorithmes d'extractions de motifs séquentiels sur les flots de données ne capturent généralement pas toute la richesse de ces flots puisqu'ils travaillent uniquement sur une seule dimension d'analyse. Ainsi, il n'existe à l'heure actuelle et à notre connaissance qu'une seule approche [PCL+05] 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. Or, d'un point de vue décisionnel, il est généralement plus intéressant d'extraire des motifs en se basant sur le maximum de dimensions d'analyses possibles. La question à laquelle nous souhaitons répondre dans ce chapitre est la suivante : Quid des flots qui, en pratique, sont multidimensionnels ?

Dans ce chapitre, nous étendons l'approche multidimensionnelle de [PCL+05] et proposons ainsi une nouvelle méthode d'extraction des motifs séquentiels multidimensionnels ou multi-attributs sur les flots de données.

Une partie des travaux présentés dans ce chapitre a été publiée dans la conférence *10th International Conference on Data Warehousing and Knowledge Discovery (DAWAK 2008)*.

1 Introduction

Comme nous l'avons vu précédemment, les motifs séquentiels sont de plus en plus utilisés dans des cadres applicatifs industriels tels que la détection de comportement des utilisateurs [SCDT00], l'analyse des logs Webs [PHMaZ00], l'analyse des séquences de protéines [YWYH02] ou l'analyse de séquence musicale [HLC01]. Cependant, comme nous l'avons vu au cours de ce mémoire, les propositions existantes ne travaillent que sur une seule dimension d'analyse, généralement appelée *produit* dans l'*étude du panier de la ménagère*.. Bien que la littérature recense des contributions liées aux motifs séquentiels multidimensionnels introduits par [PHP+01], celles-ci ne permettent pas d'extraire des motifs combinant plusieurs attributs. De plus, avec le progrès constant dans les technologies logicielles et matérielles, il devient de plus en plus aisé pour les entreprises et les organisations de générer, récupérer et stocker les informations provenant de plusieurs sources telles que les réseaux TCP/IP, les transactions financières issues des opérations faites avec les cartes bancaires, les informations médicales ou même les réseaux de capteurs. Ce genre de données collectée pousse les chercheurs à étudier le nouveau modèle des flots de données et à se pencher vers les différentes approches de représentations du temps comme dans [MM02, GHP+03, CWYM04, LLS04, TCY03] qui introduisent ou étendent les différentes approches dites du *landmark*, *des fenêtres glissantes* ou des modèles basés sur *l'oubli temporel*. Comme nous l'avons vu précédemment, sur le peu d'approches existantes d'extractions de motifs séquentiels sur le flot [CWZ05, MM06, RPT06], pas une *seule* n'utilise plus d'une dimension d'analyse.

Dans ce chapitre, nous introduisons une approche d'extraction de motifs séquentiels multidimensionnels qui permet d'extraire de nouvelles connaissances permettant des analyses de tendances et des détections de connaissances atypiques. Malheureusement, bien que ces connaissances soient d'un avantage certain pour les analystes et les décideurs, extraire tous les motifs séquentiels multidimensionnels est impossible à cause de l'espace de recherche énorme qui est associé à ce problème d'énumération. Afin de résoudre ce problème, nous nous intéressons à une classe spécifique de ces motifs : les motifs séquentiels multidimensionnels construits à partir des items multidimensionnels les plus spécifiques. De plus, afin de gérer le processus d'historisation nécessaire à n'importe quel système de requêtes sur les résultats d'extraction, nous utilisons les tables de fenêtres temporelles (C.f. Chapitre 6) qui, comme nous le démontrons plus tard, est le modèle le plus adapté pour ce genre de problématique.

Les deux contributions majeures dans ce chapitre sont :

1. Un algorithme d'extraction de motifs séquentiels multidimensionnels sur un flot de données. Les motifs sont mis à jour dynamiquement au moyen des tables de fenêtres temporelles. Le stockage des motifs se fait au moyen d'une structure d'arbre préfixé. Cette structure de donnée permet de résumer les supports des motifs sur plusieurs niveaux de granularité temporelle.

2. La possibilité de détecter automatiquement la généralisation et la spécialisation des items multidimensionnels ou des séquences à travers le temps. Dans le cadre de notre approche, il devient ainsi possible pour un analyste d'extraire les motifs et de suivre leurs évolutions sur le flot en terme de spécialisation, de généralisation et de fréquences. Ce type de connaissances permet ainsi d'induire des connaissances d'un niveau d'abstraction plus haut telles que l'analyse de tendances ou même la détection de connaissances atypiques (*outliers*).

Le chapitre est organisé de la manière suivante. La section 2 présente les différents travaux connexes à notre problématique. Les concepts préliminaires sont proposés dans la section 3. Dans la section 4, nous introduisons la problématique avec un exemple illustrant l'extraction de motifs séquentiels multidimensionnels sur un flot de données. Notre algorithme est présenté dans la section 5. Les expériences réalisées sont décrites et discutées dans la section 6. Nous concluons ce chapitre par une discussion.

2 Travaux antérieurs

La problématique est fortement liée à l'extraction de motifs et à la prise en compte des flots. Ces deux points ayant été traités précédemment (Cf. Chapitre 2 et 6), dans cette section, nous nous focalisons plus particulièrement sur les travaux d'extraction de motifs dans un cadre multidimensionnel.

Récemment, les motifs séquentiels ont été étendus à l'extraction de motifs séquentiels multidimensionnels [PHP+01], [YC05] et [PCL+05]. Les travaux de [PHP+01] sont les premiers à considérer plusieurs dimensions dans le contexte des motifs séquentiels. Cependant, les séquences extraites par cette approche ne contiennent pas plusieurs dimensions. Généralement, les autres dimensions sont *statiques* et ne sont nullement utilisées dans le processus de fouille. Dans [YC05], les auteurs cherchent à extraire des motifs séquentiels dans le contexte du *Web Usage Mining* en considérant uniquement 3 dimensions d'analyses (pages, sessions et jours). Cependant, les séquences extraites peuvent être largement biaisées car elles décrivent des corrélations entre des attributs multidimensionnels sur une seule dimension temporelles.

Dans [PCL+05], les séquences extraites permettent une vraie combinaison de plusieurs dimensions d'analyses en prenant en compte les dimensions temporelles. Ainsi la séquence *Un client qui achète une planche de surf et une housse à New York, achète plus tard une combinaison à San-Francisco*, on remarque bien que *New York* apparaît avant *San Francisco* mais aussi que le produit *Planche de surf* précède le produit *Combinaison*.

Peu de travaux se sont focalisés sur l'extraction de données multidimensionnelles sur les flots. Ceci est généralement dû à la complexité de la tâche. Dans [HCD+05], les auteurs introduisent un algorithme d'extraction de cubes multidimensionnels sur les flots qui permet de calculer différents

niveaux (observations et X). Mais aucune approche n'a été proposée dans le cadre de la fouille de motifs séquentiels multidimensionnels.

Dans les travaux présentés dans [CWZ05], les auteurs introduisent le concept de séquences multidimensionnelles. Cependant dans cette approche, sémantiquement différente de la nôtre, les auteurs considèrent chaque flot comme une dimension d'analyse et travaillent sur l'extraction de séquences sur plusieurs flots (d'où la multidimensionnalité présumée). Notre approche est fondamentalement différente puisque c'est le flot en soi qui est considéré comme multidimensionnel. Cette définition nous apparaît plus naturelle et plus en adéquation avec la réalité des applications générant ou utilisant des flots.

3 Concepts préliminaires

Dans cette section, nous rappelons les définitions issues de [PCL+05] dans le cadre de l'extraction de motifs multidimensionnels dans une base de données statique. Puis nous étendons ces définitions dans le cadre des flots de données.

3.1 Motifs séquentiels multidimensionnels

Données manipulées

Nous étendons les concepts présentés dans le chapitre 2 : séquence, date de transaction et d'items en considérant non plus des attributs simples pour décrire les données, mais des ensembles finis d'attributs. Nous supposons qu'il existe au moins une dimension (*e.g.* temporelle) dont le domaine est totalement ordonné.

Définition 8.1 (Partition des dimensions)**.** Pour chaque table définie sur un ensemble de dimensions D, nous considérons une partition de D en quatre sous-ensembles notés respectivement :

- D_R pour l'ensemble des dimensions de référence (identifiant de séquence ou 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 sur lesquelles les corrélations seront extraites (items ou produits dans le contexte classique).
- D_I pour l'ensemble des dimensions ignorées.

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

Définition 1 (Bloc)**.** Etant donnée une base de données $\mathcal{D}$, on appelle bloc l'ensemble des n-uplets de $\mathcal{D}$ qui ont la même restriction r sur D_R.

Chaque bloc $\mathcal{B}_r$ est identifié par le tuple r qui le défini. On note $\mathcal{B}_{\mathcal{D},D_R}$, l'ensemble des blocs de $\mathcal{D}$ définis à partir des dimensions de référence D_R.

Exemple 8.1. Nous considérons la partition suivante des dimensions de la base de données $\mathcal{D}$ représentée sur le tableau 8.1 :

- $D_I = \emptyset$
- $D_R = \{Client\}$
- $D_A = \{Ville, Travail, Age,\ Produit\}$
- $D_T = \{Date\}$

Il est possible de diviser $\mathcal{D}$ en trois blocs selon les restrictions sur les valeurs pour la dimension *Client*. Ces trois blocs définissent une partition en bloc de $\mathcal{D}$-*Octobre* (Figure 3.1).

$\mathcal{D}$-*Octobre*					
Clients	Date	Informations clients			
C_1	1	*NY*	*Educ.*	*Moyen*	*CD*
C_1	1	*NY*	*Educ.*	*Moyen*	*DVD*
C_1	2	*LA*	*Educ*	*Moyen*	*CD*
C_2	1	*SF*	*Prof.*	*Moyen*	*PS*3
C_2	2	*SF*	*Prof.*	*Moyen*	*Xbox*
C_3	1	*DC*	*Commercial*	*Senior*	*PS*2
C_3	1	*LA*	*Commercial*	*Senior*	*Jeu*

Tab. 8.1: Une table multidimensionnelles $\mathcal{D}$

Client	Date	Informations clients			
C_1	1	*NY*	*Educ.*	*Moyen*	*CD*
C_1	1	*NY*	*Educ.*	*Moyen*	*DVD*
C_1	2	*LA*	*Educ*	*Moyen*	*CD*

Client	Date	Informations clients			
C_2	1	*SF*	*Prof.*	*Moyen*	*PS*3
C_2	2	*SF*	*Prof.*	*Moyen*	*Xbox*

Client	Date	Informations clients			
C_3	1	*DC*	*Commercial*	*Senior*	*PS*2
C_3	1	*LA*	*Commercial*	*Senior*	*Jeu*

Fig. 8.1: Les différents blocs de $\mathcal{D}$-*Octobre*

- L'ensemble D_R permet d'énumérer pour le calcul du support des séquences multidimensionnelles. Dans le cadre de l'extraction de motifs séquentiels classique, la cardinalité de cet

ensemble est égale à un (généralement l'attribut identifiant du client ou l'identifiant de la séquence).
- L'ensemble D_A décrit les dimensions d'*analyse*, les attributs présents dans les motifs séquentiels multidimensionnels extraits viendront de cet ensemble de dimensions. Comme pour D_R et dans le cadre des motifs classiques, une seule dimension est utilisée (comme par exemple les produits achetés dans le cadre du *panier de la ménagère*).
- L'ensemble D_I décrit les dimensions *ignorées* lors du processus d'extractions. Les attributs présents dans cet ensemble ne seront donc pas présents dans les motifs séquentiels extraits.

3.2 Item, itemset et séquence multidimensionnels

Comme dans le cadre d'extraction de motifs séquentiels classique, il existe des définitions pour les concepts d'item, itemset et séquence multidimensionnels. Nous rappelons ici les différentes définitions présentés dans [PCL+05].

Définition 2 (Item multidimensionnel)**.** Un item multidimensionnel $e = (d_1, \ldots, d_m)$ est un m-uplet défini sur D_A tel que :
- $\forall i \in [1, \ldots m],\ d_i \in Dom(D_i) \cup \{*\}$
- $\exists d_i \in e$ t.q. $d_i \neq *$

Exemple 8.1. Dans $\mathcal{D}$-*Octobre*, $(NY, Educ, Moyen, CD)$, $(DC, Commercial, Senior, PS2)$ et $(*, Commercial, Senior, Jeu)$ sont des items multidimensionnels.

Il est généralement impossible (dans les bases de données réelles) de trouver une instantiation sur l'intégralité des dimensions d'analyse. Afin de palier ce problème d'instanciation, dans [PCL+05], les auteurs introduisent une valeur *joker* symbolisée par $*$. Cette valeur permet d'ignorer les valeurs sur la dimension d'analyse ciblée. Dans le cadre d'une extraction de motifs, n'importe quel item multidimensionnel traité doit contenir au moins une dimension d'analyse spécifiée.

Définition 8.2 (Itemset multidimensionnel)**.** Un itemset multidimensionnel $i = \{e_1, e_2, \ldots, e_p\}$ est un ensemble non-vide d'items multidimensionnels.

Exemple 8.2. Dans $\mathcal{D}$-*Octobre*, $\{(NY, Educ, Moyen, CD), (NY, Educ, Moyen, DVD)\}$ et $\{(DC, Commercial, Senior, PS2), (LA, Commercial, Senior, PS2)\}$ sont des itemsets multidimensionnels.

Définition 8.3 (Séquence multidimensionnelle)**.** Une séquence multidimensionnelle $\varsigma = \langle i_1, \ldots, i_l \rangle$ est une liste ordonnée non vide d'itemsets multidimensionnels.

Exemple 8.3. Dans $\mathcal{D}$-*Octobre*, $\langle\ \{(NY, Educ, Moyen, CD), (NY, Educ, Moyen, DVD)\}$ $\{(LA, Educ, Moyen, CD)\}\ \rangle$ est une séquence multidimensionnelle.

Définition 8.4 (Inclusion de séquence)**.** Une séquence multidimensionnelle $\alpha = \langle a_1, \ldots, a_l \rangle$ est une sous-séquence de $\gamma = \langle b_1, \ldots, b_{l'} \rangle$, notée $\alpha \prec \gamma$, si et seulement si il existe des entiers $1 \leqslant j_1 \leqslant j_2 \leqslant \ldots \leqslant j_l \leqslant l'$ tels que $a_1 \subseteq b_{j_1}, a_2 \subseteq b_{j_2}, \ldots, a_l \subseteq b_{j_l}$.
La séquence γ est dites super-séquence de α.

Exemple 8.4. Soient les séquences multidimensionnelles
$\alpha = \langle \{(NY, Educ, Moyen, CD), (NY, Educ, Moyen, DVD)\} \rangle$ et
$\gamma = \langle \{(*, *, Moyen, CD), (NY, Educ, Moyen, *)\} \{(LA, Educ, Moyen, CD)\} \rangle$, α est une sous-séquence de γ.

3.3 Support

Calculer le support d'une séquence revient à compter le nombre de blocs de la base qui la supportent. Un bloc supporte une séquence si on peut trouver un ensemble d'ensembles de cellules contenant les itemsets de la séquence tout en respectant la relation d'ordre imposée par la séquence. La présence d'une valeur joker $*$ sur une dimension permet la relaxation d'une contrainte sur les attributs de ladite dimension.

Exemple 8.5. Dans $\mathcal{D}$-*Octobre*, la séquence $\langle \{(NY, Educ, Moyen, CD), (NY, Educ, Moyen, DVD)\}$ $\{(LA, Educ, Moyen, CD)\} \rangle$ à un support de 1 puisqu'elle est incluse uniquement dans le bloc B_{C_1}.

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 valeur $*$ dans un item de la séquence, l'inclusion (definition 8.4) 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.

4 Extraction de motifs séquentiels multidimensionnels sur les flots de données

Dans notre approche nous supposons que le flot de données peut être représenté comme une séquence ordonnée contenant une infinité de *batches* et peut s'exprimer comme $DS = \langle B_0, B_1, \ldots, B_n \rangle$. Un batch B_i est un ensemble de blocs multidimensionnels définis préalablement sur un ensemble fini de dimensions D apparaissant sur le flot de données $B_i = \{\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3, \ldots, \mathcal{B}_k\}$. Chaque batch est de cardinalité fixe en termes de transactions dans les différents blocs. Bien qu'il soit possible de définir un batch sur une fenêtre temporelle nous proposons l'utilisation basée sur la taille et ceci afin d'éviter que notre approche ne soit dépendante en terme d'efficacité de la distribution des blocs multidimensionnels apparaissant sur le flot et qui est généralement supposée inconnue. De plus, chaque batch est associé à une estampille temporelle logique t, i.e. B_t, et nous supposerons

que l'identifiant n est l'identifiant du batch B_n le plus récent. La taille du flot de données peut être définie comme : $L_{DS} = |B_0|+|B_1|+\ldots+|B_n|$ où $|B_i|$ est le cardinal du batch i en terme de blocs multidimensionnels. Dans l'exemple présenté dans la figure 8.2, $L_{DS} = |B_{\mathcal{D}\text{-}Octobre}|+|B_{\mathcal{D}\text{-}Novembre}| = 7$ avec $|B_{\mathcal{D}\text{-}Octobre}| = 3$ ($\mathcal{B}_{C_1}$, $\mathcal{B}_{C_2}$ et $\mathcal{B}_{C_3}$) et $|B_{\mathcal{D}\text{-}Novembre}| = 4$ ($\mathcal{B}_{C_4}$, $\mathcal{B}_{C_5}$,$\mathcal{B}_{C_6}$ et $\mathcal{B}_{C_7}$). La taille des 2 batches en terme de transactions est de 7.

$\mathcal{D}$-*Octobre*						$\mathcal{D}$-*Novembre*					
Clients	Date	Informations clients				*Clients*	Date	Informations clients			
C_1	1	*NY*	*Educ.*	*Moyen*	*CD*	C_4	1	*NY*	*Comm.*	*Moyen*	*Wii*
C_1	1	*NY*	*Educ.*	*Moyen*	*DVD*	C_4	2	*NY*	*Comm.*	*Moyen*	*Jeu*
C_1	2	*LA*	*Educ*	*Moyen*	*CD*	C_5	1	*LA*	*Prof.*	*Moyen*	*Wii*
C_2	1	*SF*	*Prof.*	*Moyen*	*PS*3	C_5	1	*LA*	*Prof.*	*Moyen*	*iPod*
C_2	2	*SF*	*Prof.*	*Moyen*	*Xbox*	C_6	1	*LA*	*Educ.*	*Junior*	*PSP*
C_3	1	*DC*	*Comm.*	*Senior*	*PS*2	C_6	2	*LA*	*Educ.*	*Junior*	*iPod*
C_3	1	*LA*	*Comm.*	*Senior*	*Jeu*	C_7	1	*LA*	*Comm.*	*Junior*	*PS*2

Fig. 8.2: Exemple de 2 batches apparaissant sur un flot de données multidimensionnel

A partir de ces définitions du flot de données et des batches, l'extraction des motifs séquentiels multidimensionnels revient à énumérer l'ensemble des séquences multidimensionnelles issues du flot et respectant la contrainte suivante :

$$Support(S) \geqslant \sigma.L_{DS} \tag{8.1}$$

Avec σ, la valeur du support minimal choisie par l'utilisateur.

Malheureusement, dans le cadre de l'extraction sur des flots certains éléments peuvent devenir fréquents sur un laps de temps t avant de redevenir non-fréquents à l'instant $t+1$ et vice-versa. La définition 8.1 est donc trop contraignante puisqu'elle ne prend pas en compte ce genre de scénario. Nous devons donc introduire plus de souplesse dans la contrainte d'extraction et permettre dans notre approche le stockage de séquences qui pourraient devenir non-fréquentes puis fréquentes ou vice-versa. Notre approche devra donc stocker les motifs séquentiels fréquents en plus d'une bordure de motifs dits *sous-fréquents* en respectant la nouvelle contrainte :

$$Support(S) \geqslant \epsilon.L_{DS} \tag{8.2}$$

Avec $\epsilon < \sigma$, la valeur de sous-fréquence choisie par l'utilisateur.

Un motif séquentiel s sera dit fréquent si et seulement si : $support(s) \geqslant \sigma.L_{DS}$ et sera dit sous-fréquent si et seulement si : $\sigma.L_{DS} > support(S) \geqslant \epsilon.L_{DS}$.

En dépit de la simplicité de ces contraites et comme généralement pour toutes les applications d'extraction de connaissances sur les flots, il est impossible de stocker toutes les valeurs de support des séquences pour chaque batch en mémoire ou même sur un espace disque. Il faut donc un processus de stockage et de journalisation permettant un compromis entre la précision et l'espace disponible. Dans notre approche nous utilisons les tables de fenêtres temporelles logarithmiques[GHP+03].

4.1 Un exemple : l'analyste et les magasins de jeux vidéos

Afin d'illustrer les définitions précédentes, considérons un exemple d'extraction de motifs multidimensionnels en se basant sur les 2 batches présentés dans la figure 8.2 et qui contiennent les achats de clients dans différents magasins d'une entreprise de jeux. Plus précisément, ces batches décrivent les achats des clients selon 6 dimensions : la dimension *Client* qui contient l'identifiant du client (généralement le numéro de carte bancaire ou le numéro de fidelité), la dimension *date*, la dimension *Ville* qui permet la localisation géographique des clients, la dimension *Travail* qui permet de classer les clients selon les différentes classes socio-professionelles, la dimension *Age* et enfin la dimension *Produit* qui représente les différents achats du client à une date donnée.

Supposons maintenant qu'un analyste de l'entreprise de jeux décide de lancer une extraction de motifs séquentiels dans le but d'extraire de nouvelles connaissances sur l'évolution des achats des utilisateurs sur la période d'Octobre-Novembre afin de planifier les prochaines politiques de marketing.

L'analyste extrait donc toutes les séquences qui apparaissent chez au moins 50% des clients dans chaque batch. Du batch du mois d'Octobre, l'analyste découvre 3 différentes séquences :

1. La première séquence $\langle\{(*, *, Moyen, *)\}\rangle$ qui est une séquence contenant un seul item. Cette séquence énonce qu'au moins 2 clients sur 3 qui ont acheté des produits dans les magasins de jeux sont d'âge moyen.
2. La seconde séquence $\langle\{(LA, *, *, *)\}$ qui est aussi une séquence d'un seul item peut être traduite sémantiquement comme : 2 clients sur 3 ont fait leurs achats dans le magasin situé à Los Angeles.
3. La troisième séquence $\langle\{(*, *, Moyen, *)\}\{(*, *, Moyen, *)\}\rangle$ est une séquence contenant 2 items. Cette séquence est d'un intérêt particulier pour l'analyste puisqu'elle lui permet d'inférer que 2 clients sur 3 sont d'âge moyen et qu'ils ont fait 2 achats courant du mois d'Octobre.

Puis l'analyste extrait les séquences pour le mois de Novembre. Il y a au total 9 séquences fréquentes mais seulement 3 sont vraiment intéressantes :

1. $\langle\{(LA, *, Junior, *)\}\rangle$, cette séquence informe l'analyste que 2 clients sur 4 qui ont fait leurs achats à Los Angeles sont de jeunes personnes ($Junior$).
2. $\langle\{(LA, *, *, iPod)\}\rangle$, informe l'analyste que 2 clients sur 4 ont acheté un produit $iPod$ à Los Angeles. L'analyste peut à partir de cette connaissance proposer une offre promotionnelle pour le prochain mois qui prendrait en compte la localisation géographique et le produit $iPod$. Il faut également remarquer que cette séquence est une spécialisation de la deuxième séquence extraite du batch du mois d'Octobre. Ceci veut dire qu'au mois d'Octobre 2 clients sur 3 faisaient leurs achats dans le magasin de Los Angeles mais qu'aucun produit ne se détachait vraiment (d'où le $*$ sur la dimension *Produit*) et que pour le mois de Novembre cette tendance s'est affirmée pour le produit $iPod$.
3. $\langle\{(*, *, Moyen, Wii)\}\rangle$, cette séquence permet d'inférer que 2 clients sur 4 sont d'âge moyen et ont acheté la console Wii en Novembre. Cette séquence est aussi une spécialisation de la première séquence du batch associée au mois d'Octobre. Cette séquence permet à l'analyste de détecter l'apparition d'une nouvelle tendance avec l'arrivée sur le marché des nouvelles consoles Wii. L'analyste peut donc coupler ces connaissances pour proposer par exemple de nouvelles promotions comme un iPod à moitié prix pour chaque console Wii acheté.

Grâce à cet exemple, nous avons présenté quelques-uns des principaux intérêts de l'extraction de motifs séquentiels sur les flots de données : la détection de tendance de haut niveau entre les différentes séquences (e.g. la spécialisation ou la généralisation de la séquence), la détection de nouvelles séquences contenant de nouveaux attributs et la détection de disparitions de tendances ou de séquences.

L'algorithme $MDSDS$ que nous proposons permet de répondre à ce genre de requêtes. Pour cela, notre algorithme maintient en mémoire une structure de données qui permet l'historisation et le resumé des informations sur les séquences fréquentes et sous-fréquentes.

5 L'approche $MDSDS$

Dans $MDSDS$, l'extraction de motifs séquentiels multidimensionnels est l'étape la plus difficile. De manière à l'optimiser, nous divisons le processus en deux tâches :

1. $MDSDS$ extrait les items multidimensionnels les plus spécifiques. Cette politique permet de limiter l'espace de recherche qui est potentiellement très grand. De plus, le choix des items multidimensionnels les plus spécifiques permet de *factoriser* les connaissansaces, puisque les motifs plus généraux peuvent être inférés en une étape de post-traitement. Le maintien des items multidimensionnels les plus spécifiques permet aussi de détecter les spécialisations et généralisations des séquences multidimensionnelles en repérant l'apparition de la valeur $*$ sur les prochains items des batchs suivants sur le flot de données.

2. L'extraction des motifs séquentiels multidimensionnels se fait en utilisant PrefixSpan[PHMa+04] et en se basant uniquement sur les items multidimensionnels les plus spécifiques précédemment extraits.

L'application de cette stratégie de découverte en 2 étapes permet d'éviter l'extraction des séquences multidimensionnelles trop générales. Ces séquences ne sont pas intéressantes du point de vue de l'analyste parce qu'elles contiennent de la connaissance *"trop vague"* qui est généralement inutilisable dans la prise de décision.

$MDSDS$ utilise une structure de données utilisant un arbre prefixé contenant dans ses noeuds les items et leurs tables de fenêtres temporelles associées (cf. Figure 8.3) afin de journaliser les informations sur les séquences fréquentes et sous-fréquentes.

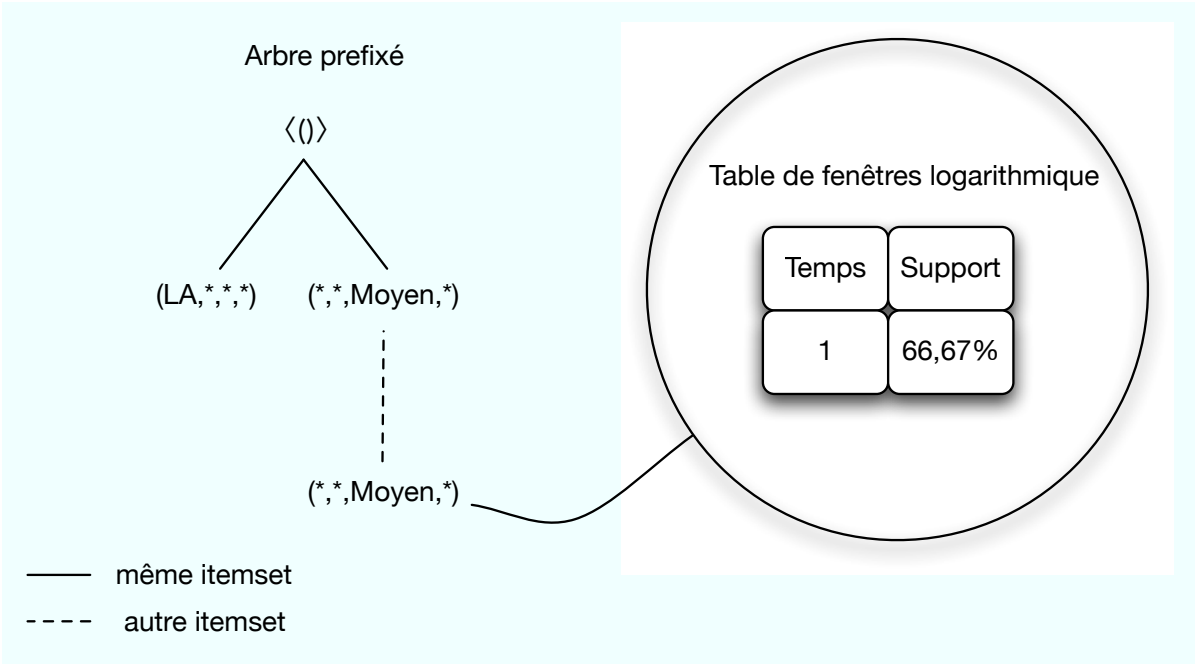


Fig. 8.3: L'arbre prefixé pour le batch du mois d'Octobre

Les motifs séquentiels dans notre approche sont divisés en 3 catégories :

1. Les motifs fréquents qui sont stockés dans l'arbre prefixé.
2. Les motifs sous-fréquents qui sont des motifs qui pourraient devenir potentiellement fréquents dans les prochains batchs. $MDSDS$ doit donc les stoker tout comme les motifs séquentiels fréquents.
3. Les motifs infréquents ne sont pas stockés. Lorsqu'un motif sous-fréquent devient infréquent (son support devient plus petit que ϵ) il est automatiquement élagué.

Les opérations de mise à jour se font après avoir reçu le batch du flot : au niveau de l'arbre prefixé 2 opérations sont effectuées, tout d'abord rajouter les nouveaux items en tant que nœud

dans l'arbre, puis mettre à jour les tables de fenêtres temporelles en introduisant uniquement la valeur du support de la séquence dans le batch actuel. Si nécessaire, $MDSDS$ peut lancer une opération d'élagage sur les tables de fenêtres temporelles en respectant les contraintes presentées dans [GHP+03].

5.1 Algorithme

Nous allons maintenant décrire plus en détail les différentes étapes de notre algorithme $MDSDS$ presenté dans Algorithme 6.

Algorithme 6 : Algorithme $MDSDS$.

Data : Support minimum σ, seuil de sous-fréquence ϵ, batch B_i

Result : L'ensembles des motifs séquentiels multidimensionneles fréquents et sous-fréquents sur DS

```
1 begin
2   foreach batch B_i ∈ DS do
      /* Extraire les items multidimensionnels les plus spécifiques */
3     SI_i ← MINE_ITEMS(B_i, ε);
      /* Extraire tous les motifs séquentiels multidimensionnels à partir de
         B_i */
4     MINE_SEQUENCES(ε);
      /* Mise à jour de l'arbre prefixé et des tables de fenêtres temporelles
         */
5     SCAN_TREE();
6 end
```

Comme nous l'avons vu précédemment, la tâche la plus importante est l'extraction d'items multidimensionnels les plus spécifiques. Afin d'optimiser ce processus, nous ramenons ce problème à la problématique d'extraction d'itemsets maximaux fréquents. Ainsi chaque valeur dans chaque dimension d'analyse est associée à un entier que l'on appellera *entier associé*. Cette approche est possible si et seulement si la contrainte suivante est respectée pour les batches en entrée de l'algorithme :

$$\bigcap_{\mathcal{I}_i \in D_A} = \emptyset \tag{8.3}$$

Avec $\mathcal{I}_i$ l'ensemble des entiers associés pour les différentes valeurs possibles dans la $i^{\text{ème}}$ dimension d'analyse.

Cette contrainte stipule que 2 valeurs différentes sur 2 dimensions d'analyses différentes ne peuvent pas avoir le même entier associé. Illustrons cette technique avec l'exemple du batch du mois d'Octobre de la figure 8.2. La valeur *New York* dans la dimension d'analyse *Ville* aura l'entier associé 1, la valeur *Los Angeles* de la même dimension aura l'entier associé 2 et ainsi de suite jusqu'à la valeur *Jeu* de la dimension d'analyse *Produit* qui aura l'entier associé 15 (voir Table 8.2).

$\mathcal{D}$-*Octobre*					
Clients	Date	Informations clients			
C_1	1	1	5	8	10
C_1	1	1	5	8	11
C_1	2	2	5	8	10
C_2	1	3	6	8	12
C_2	2	3	6	8	13
C_3	1	4	7	9	14
C_3	1	2	7	9	15

Tab. 8.2: Le batch $\mathcal{D}$-*Octobre* après transformation en entrée pour l'algorithme *MDSDS*.

Ainsi en utilisant un algorithme de fouille d'itemsets sur cette représentation du batch du mois d'Octobre et en supposant que le support minimal $\sigma = 0.5$, on extrait les 2 itemsets maximaux suivants : (2) et (8) (de support 2) qui sont de fait les 2 items multidimensionnels les plus spécifiques : (Los Angeles, $*, *, *$) et ($*, *$, Moyen, $*$). Dans ce cas, le support des itemsets est défini selon le nombre de blocs qui contiennent l'itemset.

Par souci de clarté, supposons maintenant que le client C_2 ait un travail *Educ.* et non plus *Prof.*, on a alors le bloc présenté dans la Table 8.3

Client	Date	Informations clients			
C_2	1	3	5	8	12
C_2	2	3	5	8	13

Tab. 8.3: Bloc du client C_2 dans le batch $\mathcal{D}$-*Octobre* après changement de la valeur dans la dimension d'analyse *Travail*.

Avec ce léger changement, on extrait les 2 nouveaux itemsets maximaux suivants : (2) et (5, 8) qui désignent les 2 items multidimensionnels les plus spécifiques : (Los Angeles, ∗, ∗, ∗) et (∗, Educ, Moyen, ∗).

Afin d'extraire les motifs fréquents et sous-fréquents, l'algorithme $MDSDS$ appelle la procédure $MINE\_SEQUENCES(\epsilon)$. Le support minimal est bien sûr fixé sur le seuil des motifs sous-fréquents ϵ et l'algorithme utilise en sous-procédure l'algorithme PrefixSpan [PHMa+04] afin d'extraire les séquences. Les nouvelles séquences sont immédiatement rajoutées à l'arbre prefixé, celles déjà présentes dans l'arbre, du fait de leur extraction dans des batches précédents, sont mises à jours lors de l'appel de la procédure $SCAN\_TREE$ qui met à jour chaque table de fenêtres temporelles pour chacune des séquences présentes dans l'arbre comme défini dans [GHP+03].

6 Expérimentations

Dans cette section nous présentons les différents résultats des expérimentations que nous avons conduits afin de prouver la faisabilité de notre approche $MDSDS$. Au travers de cette section nous essayons de répondre aux différentes questions :

1. *Est-ce que l'algorithme extrait et met à jour ses structures à temps avant l'arrivée du prochain batch ?*
2. *Est-ce que le processus de fouille sur le flot de données reste borné en terme de mémoire ?*

Nous décrivons tout d'abord la méthode expérimentale puis nous analysons et discutons les résultats.

6.1 Méthode expérimentale

Nous avons réalisé les expérimentations sur un ordinateur Core-Duo 2.16 Ghz MacBook Pro avec 2Go de mémoire tournant sous Mac OS X 10.5.2. L'algorithme $MDSDS$ à été écrit en C++ en utilisant une version modifié du code Apriori[1] afin d'extraire les items multidimensionnels les plus spécifiques et nous avons modifié l'implémentation de PrefixSpan[2] afin de prendre en compte la fouille de séquences multidimensionnelles.

Les expérimentations ont été faites sur des jeux de données réelles issues d'une capture en temps réel du trafic TCP/IP du Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier. Nous limitons la taille des batches à 20000 transactions et une moyenne de 5369 blocs

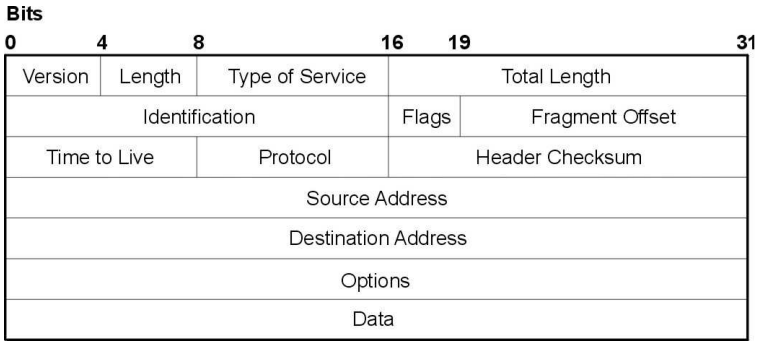
[1] http ://www.adrem.ua.ac.be/ goethals/software/

[2] http ://illimine.cs.uiuc.edu/

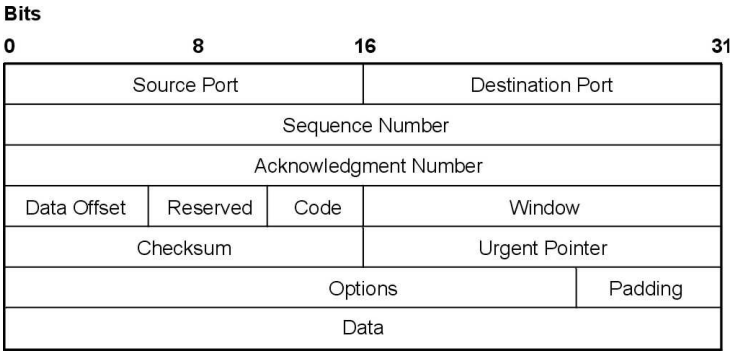
par batch. Le temps de remplissage des batches varie selon la distribution du trafic (le trafic est beaucoup moins volumineux la nuit par exemple).

Un réseau TCP/IP peut être vu comme un flot multidimensionnel. Ceci peut s'expliquer de la manière suivane : chaque en-tête TCP et IP contient une multitude d'informations encapsulées dans plusieurs formats différents appelés champs (Cf. Figure 8.4). Ces champs contiennent par exemple l'adresse source du paquet, sa destination, son numéro de séquence, certaines options spécifiques au protocole TCP/IP comme la fragmentation des paquets, la taille de la fenêtre etc...

Dans nos expérimentations, nous avons décidé de prendre en compte cette richesse du flot et nous considérons certaines parties d'un paquet TCP/IP comme des dimensions d'analyses. Ces informations peuvent varier drastiquement selon la destination du paquet, de l'application utilisée, les données qu'il contient ou même de la politique de securité appliquée sur le réseau. Dans cette section, nous allons montrer que notre approche peut détecter la généralisation et la spécialisation des paquets réseaux. Nous montrerons que ce genre de tendances peut avoir pour origines plusieurs facteurs comme les dénis de services, l'évolution des besoins des utilisateurs sur un réseau au cours du temps ou même l'évolution des applications.



(a) En-tête IP

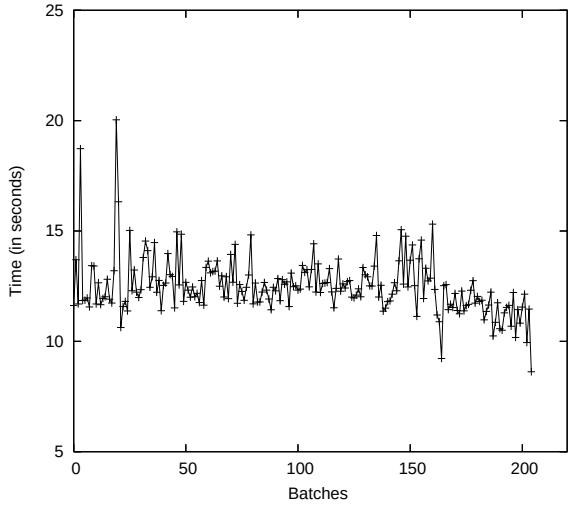


(b) En-tête TCP

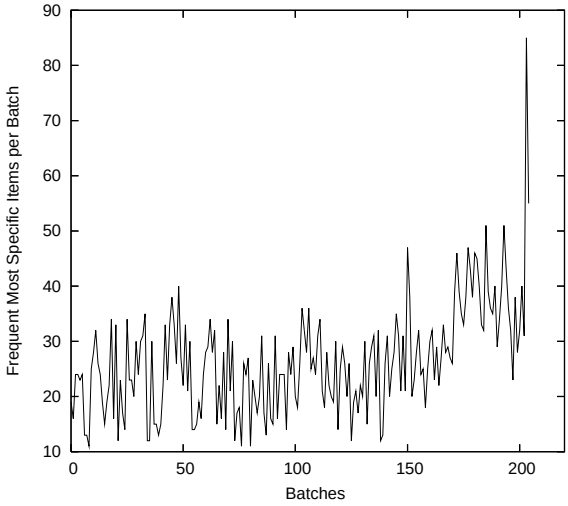
Fig. 8.4: Les champs des paquets TCP/IP pouvant être utilisés comme dimensions d'analyses

Notre jeu de données, que nous noterons $DS1$, représente un flot de données multidimensionnel très dense. Nous sélectionnons sur ce flot 13 dimensions d'analyses. Le choix de ces dimensions

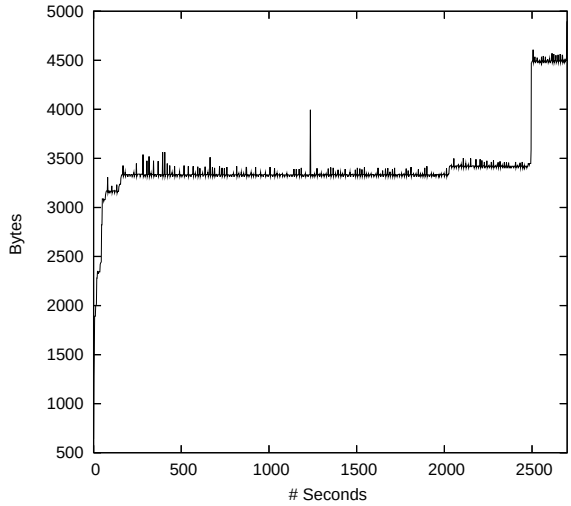
d'analyses, qui représentent pour chacune un champ de l'en-tête TCP/IP, à été validé par un expert réseau (port source, port destination, time-to-live, numéro de séquence etc...). Ce jeu de données représente un flot de données divisé pendant la capture en 204 batches pour une taille de 1.58 Go. Le temps moyen pour remplir les 20000 transactions d'un batch est de 37.3 secondes (avec une large déviation pour les heures de nuits ou la moyenne peut monter à 88.2 secondes). Les différents résultats de l'expérimentation sont présentés dans les figures 8.5(a), 8.5(b), 8.5(c) and 8.5(d).



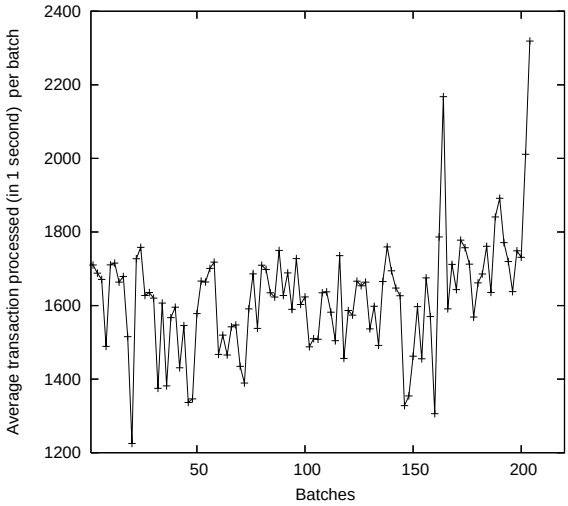
(a) Temps nécessaire pour le traitement des différents batches de $DS1$



(b) Nombre des items multidimensionnels les plus spécifiques par batch pour $DS1$



(c) Utilisation mémoire lors du traitement du flot $DS1$

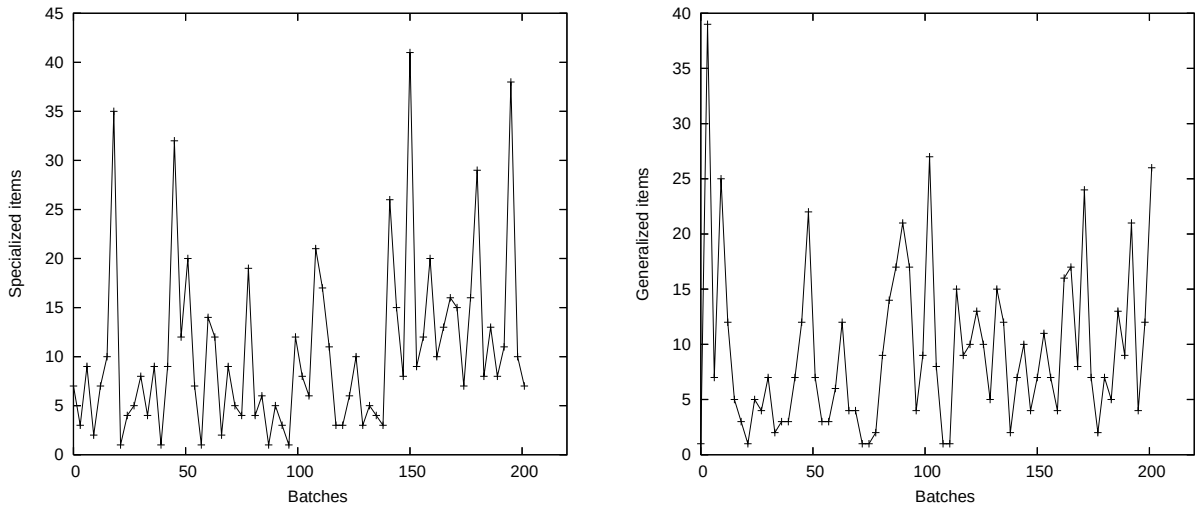


(d) Nombre de transactions moyennes traitées par seconde pour le flot de $DS1$

Fig. 8.5: Expérimentations sur les données du réseau TCP/IP

Comme on peut le voir sur la figure 8.5(a), le temps de traitement pour un batch sur $DS1$ ne dépasse pas les 25 secondes pour un support de $\sigma = 0.15\%$ et un support de sous-fréquence de

$\epsilon = 0.1\%$. Notre algorithme répond ainsi à la première contrainte : mettre à jour ses structures de données et réussir une extraction avant l'arrivée du prochain batch. Cette contrainte permet ainsi d'avoir des résultats en *temps réel*. Le temps total pour l'expérimentation est de 2600 secondes ce qui est plus bas que les 7298 secondes qui ont été nécessaires à la capture et la construction des batches pour le flot $DS1$. De plus, l'algorithme est assez performant puisqu'il traite à peu près en moyenne 1400 transactions par seconde. La figure 8.5(c) illustre l'utilisation mémoire pour le flot $DS1$. La deuxième contrainte de maîtrise de la mémoire utilisé est ici aussi respectée puisque dans le cadre de cette expérience, les structure de données utilisées dans $MDSDS$ ne dépassent pas les 6 Mo de mémoire bien qu'il y ait quelques pics de consommation mémoire dûs à la mise à jour des tables de fenêtres temporelles. Dans la figure 8.5(b), on voit le nombre d'items multidimensionnels les plus spécifiques extraits et maintenus dans la structure d'arbre prefixé de $MDSDS$, on remarque aussi que ce nombre augmente légèrement en fin d'expérimentation, soulignant une tendance vers des paquets de plus en plus spécifiques pour une application ou un utilisateur donné. Cette tendance est d'ailleurs nettement plus visible sur la figure 8.6 qui traite des différentes spécialisations et généralisations des items multidimensionnels au fur et à mesure de l'arrivée des batches.



(a) Nombre d'items qui se sont specialisés par batch **(b)** Nombre d'items qui se sont généralisés par batch

Fig. 8.6: Specialisation et generalisation des items multidimensionnels

Les spécialisations et généralisations des items multidimensionnels fréquents (Figure 8.6(a) et 8.6(b)) sont d'une importance capitale dans l'analyse des flots de données multidimensionnels puisqu'ils permettent de détecter les tendances actuelles présentes dans les séquences. Ainsi, à partir de nos résultats sur $DS1$, un expert en réseau informatique a pu extraire certaines connaissances sur l'état du trafic réseau du Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier.

Par exemple, entre le batch 150 et 151, l'item multidimensionnel $(VER : 4, IPLEN : 5, TOS : 0, PLEN > 21, IPID : [10K, 20K[, [AF], DF : 1, PROT : 6, TCPHLEN : 5, SEQ\_NUM : +65K)$ qui stipule que la plupart des paquets (selon la valeur du support) sont basés sur le procotole TCP ($PROT : 6$) avec quelques options comme la taille du paquet ($TCPHLEN : 5$) ou l'autorisation de fragmentation ($DF : 1$) se voit spécialisé en $(VER : 4, IPLEN : 5, TOS : 0, PLEN > 21, IPID : [10K, 20K[, [AF], DF : 1, TTL : 128, PROT : 6, TCPHLEN : 5, SEQ\_NUM : +65K)$. La spécialisation vient de l'apparition d'une nouvelle information qu'est le temps de survie du paquet (ou TTL) qui est ici égal à 128. Cette information permet en fait de conclure qu'au batch 150, la plupart des paquets ont des temps de survie hétérogènes (64, 128 ou 256 sauts sans qu'aucune de ces valeurs ne se démarquent vraiment) mais qu'à partir du batch 151, les sources des paquets et leurs destinations sont toutes topologiquement concentrées à moins de 128 sauts. Cette spécialisation peut s'expliquer de plusieurs manières, la plus plausible, d'après l'analyste réseau étant que pendant ce batch, des paquets ont été envoyés vers les serveurs de messageries du laboratoire (une récupération automatique des mails ou quelques envois). En effet, dans ce cas-là, lors de la construction des paquets par le système d'exploitation, la valeur de TTL minimale est choisie car l'expéditeur et le destinateur des paquets réseaux sont sur le même segment réseau. Cette connaissance est d'ailleurs double puisque seuls les systèmes d'exploitations *Windows* et Linux encodent leurs valeurs minimales de TTL à 128, les autres systèmes d'exploitations respectent quant à eux la valeur normale de 32 ou 64[1].

7 Discussion

Dans ce chapitre, nous nous sommes intéressés au problème d'extraction de motifs séquentiels multidimensionnels sur les flots de données. Nous avons ainsi proposé le premier algorithme de ce type appelé $MDSDS$. En considérant uniquement les items multidimensionnels les plus spécifiques, nous avons montré de manière empirique qu'il était possible de diminuer l'espace de recherche parcouru et que notre approche permet ainsi de détecter des tendances sur les séquences de données.

Les expérimentations lancées sur un jeu de données réel issu du réseau TCP/IP de l'Université Montpellier 2, ont montré qu'il était possible et même intéressant pour un analyste réseau d'extraire ce genre de motifs séquentiels multidimensionnels sur un flot de paquet réseaux.

Ce travail pourraient être étendu de plusieurs manières. Par exemple, l'idée d'intégrer des hiérarchies sur les dimensions d'analyses permettraient d'apporter encore plus de clarté dans le processus d'interprétation des connaissance extraites. Il serait alors possible d'extraire des connais-

[1]Pour plus d'informations, se réferer au *RFC 1122, R. Braden, Oct 89*

sances du type : *En Octobre, les ventes de consoles de jeux vidéos sont fréquentes, alors qu'en Novembre, une console précise a été vendue fréquemment : la Wii.*

En intégrant les nouvelles dimensions d'analyse, notre objectif était de montrer la puissance d'expression de nouveaux types de séquences. En effet, alors que les approches précédentes n'étaient limitées qu'à une seule dimension, nous avons pu constater notamment dans le cas de leur mise en œuvre dans des applications réelles qu'elles étaient déjà d'une grande utilité. Aujourd'hui de très nombreuses données sont disponibles sous différentes dimensions. Offrir ainsi au décideur des données plus précises offre de nouveaux potentiels qu'il sera intéressant d'analyser par la suite.

Chapitre 9

Bilan, perspectives et conclusion

On ne peut tirer de conclusion que de ce que l'on ne comprend pas.
Peter Høeg —

Depuis le début de son histoire, jamais l'Humanité n'avait généré autant de données en si peu de temps. Depuis la fin du siècle dernier, nous sommes entrés dans une nouvelle ère, dite de la société de l'information, où le monde devrait, dit-on, appartenir à ceux qui savent s'informer le mieux et le plus vite possible. Tout le problème est justement d'éviter la saturation d'informations et de savoir trouver la connaissance la plus ciblée possible. Ces connaissances sont ainsi devenue un facteur de production au même titre que les matières premières pour les entreprises et les organisations de ce monde. Afin de relever ce nouveau défi, un ensemble de méthodes et d'approches scientifiques issues de plusieurs domaines ont été proposées afin de découvrir ces *pépites de connaissances*. Ces outils sont regroupés dans un processus complet appelé *Extraction de Connaissance dans les bases de Données*.

Dans ce mémoire, nous nous sommes particulièrement intéressés à une approche particulière de l'ECD : l'extraction de motifs séquentiels.

Afin de conclure ce manuscrit, nous proposons dans un premier temps de faire le bilan et la synthèse de nos travaux avant de discuter dans un deuxième temps des perspectives associées.

1 Bilan et synthèse

Dans le cadre de cette thèse, nous nous sommes intéressés à l'extraction de motifs séquentiels sur les flots de données. Pour cela, nous avons mis en place trois angles d'attaques :

1. Extraire une représentation condensée des motifs séquentiel sur les flots

2. Passer par une étape d'échantillonnage en une passe permettant une extraction optimale des motifs séquentiels sur le flot.
3. Extraire des motifs séquentiels multidimensionnels sur des flots qui par nature sont multidimensionnels.

1.1 Les représentations condensées pour les motifs séquentiels

Nous avons présenté dans le Chapitre 3, un bref état de l'art des représentations condensées pour les itemsets fréquents ainsi que pour les motifs séquentiels. Le but de ce chapitre, outre le caractère informatif, était de comparer les différentes méthodes de représentations et de soulever la question suivante importante : *Pourquoi y en a-t-il si peu pour les motifs séquentiels ?*

Nous avons répondu de manière détaillée à cette question dans le Chapitre 4. Ainsi, nous avons montré en utilisant une définition de classe d'équivalence basée sur une relation de support entre séquences qu'il était tout simplement inutile, voir impossible d'appliquer certaines représentations condensées se basant sur une structure mathématique (la théorie des Treillis) précise. Mais nous ne nous somme pas contentés uniquement de ce résultat, nous avons poussé l'analyse de ces classes d'équivalences et leur relation avec le problème de satisfiabilité de contraintes de fréquences `SEQFREQSAT`. Ce problème déjà défini dans le cadre des itemsets est la base de la représentation par itemsets fréquents non-dérivables. Nous avons ainsi pu exhiber une méthode se basant sur une approche algébrique (la méthode d'élimination de Fourier-Motzkin) afin de dériver des règles permettant de donner une borne supérieure à la fréquence d'une séquence *uniquement à partir des fréquences de ses sous-séquences*. Ce travail est, à notre connaissance, le seul à fournir ce genre de résultats sur des motifs structurés.

A partir de ces analyses sur les représentations condensées et fort de la nouvelle définition des classes d'équivalences pour les motifs séquentiels, nous nous sommes penchés sur le concept de règles d'associations séquentielles. Ces règles ont été étudiés de manière très formelles dans le cadre des itemsets mais rarement dans le cadre des motifs structurées dont font partie les séquences. Notre travaille dans le Chapitre 5, vise à montrer qu'il est possible d'extraire des règles d'associations séquentielles en se basant sur les différentes mesures de corrélations pour les motifs séquentiels. Pour cela, nous avons introduits un nouveau motif : les *motifs conjonctifs séquentiels*. Un motif conjonctif séquentiel est en fait un ensemble de séquences incomparables entre elles (i.e. des antichaînes). Afin d'optimiser les résultats obtenues lors de l'extraction de ces règles, nous avons introduit le concept de motifs conjonctifs séquentiels non-dérivable. Ainsi, et contrairement aux motifs séquentiels, nous avons exhibé une méthode permettant d'extraire des règles bornant le support. Nous avons utilisé dans ce cas là, un principe très connu en combinatoire : le *principe*

d'inversion de Möbius. Les expérimentations présentées dans ce chapitre montrent de manière empirique que ces motifs sont intéressants et qu'ils peuvent être extraits de manière efficace.

1.2 Echantillonnage et motifs séquentiels

Dans la Partie II, Chapitre 6, nous avons présenté un état de l'art des différents algorithmes d'extraction d'itemsets fréquents et de motifs séquentiels sur les flots. Dans le Chapitre 7, nous avons introduit une approche d'échantillonnage de base de données statiques afin d'optimiser l'extraction de motifs séquentiels. Cette approche utilise le principe d'(ϵ, δ)-approximation qui permet de garantir un taux d'erreur lors de l'opération d'échantillonnage. Nous avons ensuite transposé cette (ϵ, δ)-approximation au problème d'extraction de motifs séquentiels sur les flots de données en utilisant un algorithme d'échantillonnage par réservoir biaisé. Nous avons montré de manière formelle que cet algorithme construisait bien un réservoir biaisé contenant des séquences permettant ainsi à l'utilisateur d'extraire des motifs séquentiels selon les besoins de son application finale.

1.3 Motifs séquentiels multidimensionnels et flots de données

Dans le Chapitre 8, nous avons discuté la possibilité d'extraire des motifs séquentiels multidimensionnels sur les flots. Ces motifs sont par définition plus riches en terme de connaissance que les motifs séquentiels *normaux* puisqu'ils sont extraits à partir de plusieurs dimensions d'analyses. Nous avons proposé un algorithme d'extraction de motifs multidimensionnels sur les flots de données en se basant sur les items les plus spécifiques (ou les plus spécialisés) et une utilisation d'un système de table de fenêtres logarithmiques permettant de journaliser les résultats de l'extraction pour chaque séquence. De plus, nous avons montré dans la partie expérimentale de ce chapitre que ce genre de motifs est très intéressant dans le cadre de l'analyse des réseaux TCP/IP (qui sont des flots multidimensionnels). Ce type de connaissances extraites, permet ainsi à l'analyste, ou l'administrateur réseau, de détecter des tendances entre les différents utilisateurs de son réseau ainsi que des pannes ou des anomalies.

Les différentes propositions dans ce manuscrit nous ont permis d'éclairer sous un nouveau jour la problématique d'extraction de motifs. Ces travaux ouvrent la voie vers de nombreuses améliorations et extensions que nous détaillons dans la section suivante.

2 Perspectives

Nous détaillons ici quelques perspectives qui nous tiennent à cœur. Nous discutons des perspectives à court terme visant à améliorer l'efficacité de l'extraction de motifs séquentiels dans les bases de données statiques et sur les flots de données ainsi que l'extraction de règles d'association séquentielles. Enfin, nous distinguons les perspectives à long terme qui orientent nos travaux vers d'autres types de motifs.

2.1 Améliorer l'extraction de motifs séquentiels

Nous avons vu dans le Chapitre 4, que les règles d'antimonotonie ne permettaient pas d'élaguer de manière optimale. L'utilisation de règles de dérivation d'une borne supérieure sur le support d'une fréquence permettrait alors d'élaguer plus rapidement certaines séquences sans avoir à passer par une étape de comptage. Dans l'état actuel des choses, les règles sont extraites d'une manière très peu optimisée par une méthode d'élimination de Fourier-Motzkin. Afin d'arriver au but escompté, il serait nécessaire de trouver une formule close (comme dans le cas des formules de dérivation de support pour les itemsets) permettant de calculer très vite la borne supérieure [CG05b].

2.2 Extraction de règles d'association séquentielles

Nous avons présenté dans le Chapitre 5, un nouveau motif qui permettait l'extraction de règles d'association séquentielles. Comme dans le cadre de la dérivation de règle pour les motifs séquentiels, nous n'avons toujours pas trouvé de formule close nous permettant d'éviter de passer pas une construction de matrice et son inversion à chaque génération d'une conjonction de séquence. De plus, alors que la propriété de la monotonie de la dérivabilité pour les itemsets à été prouvée [Cal03]. Cette même question dans le cadre de ces nouveaux motifs reste ouverte. La réponse à cette question et la formule close, nous assureraient la possibilité de construction d'algorithmes très efficaces pour l'extraction de motifs conjonctifs séquentiels et donc de règles d'associations séquentielles.

Ces même motifs conjonctifs séquentiels devraient pouvoir être extraits à partir de flot de données permettant ainsi non plus d'avoir de la connaissance issue du flot sous forme de séquence uniquement mais aussi sous forme de corrélation statistiques entre des conjonctions de séquences.

2.3 Échantillonnage et motifs séquentiels

Nous avons vu précédemment que les approches d'extraction de motifs sur les flots nécessitent, pour des raisons de capacités mémoire, de supprimer ou d'approximer des connaissances acquises préalablement (par exemple, une séquence qui n'est plus fréquente sur un petit intervalle de temps est effacée). Au travers de notre approche d'échantillonnage par réservoir biaisé, il n'est plus besoin d'exécuter d'algorithme d'extraction en continu mais plutôt à la demande, à partir du réservoir maintenu en mémoire, pour fournir au décideur toute la connaissance extraite du flot en lui garantissant une marge d'erreur. Les perspectives à court termes associées à ces travaux sont nombreuses. Dans un premier temps, nous voulons pouvoir borner théoriquement la taille de la liste noire. Bien qu'une borne sur la taille de la liste noire ait été exhibée de manière empirique, nous pensons que l'utilisation de certains processus stochastiques permettraient d'exhiber une borne supérieure théorique intéressante. Une autre perspective qui nous tient à cœur dans le cadre de l'échantillonnage sur les flots seraient de tester et de comparer d'autre techniques d'échantillonnage comme par exemple l'échantillonnage distinct [Gib01] ou d'autres algorithmes d'échantillonnage par réservoir tels que [GLH06].

Dans un second temps, nous souhaitons intégrer cette approche dans un outil complet de suivi de flots de données permettant ainsi de pouvoir effectuer non seulement de l'extraction mais aussi des requêtes sur les données du flot.

2.4 Motifs séquentiels multidimensionnels

Dans la discussion du Chapitre 8, nous avons déjà esquissé l'utilisation des hiérarchies comme un moyen d'améliorer la lisibilité et l'interprétation des connaissances extraites. A coté de cette perspective, nous aimerions pouvoir approximer les valeurs des dimensions quantitatives sous différents types d'agrégats pendant l'extraction de motifs séquentiels multidimensionnels. Ce type de perspective nous rapprocherait ainsi un peu plus du cadre de travail OLAP.

3 Au delà des séquences

Nous pensons que les travaux théoriques présentés dans le cadre de cette thèse peuvent être utilisés pour d'autres motifs. Nous pensons ainsi que l'utilisation de des classes d'équivalences basés sur une relation de support pour un motifs couplée à l'utilisation d'outils de la combinatoire et de l'algèbre comme les méthodes d'éliminations et d'inversions pourraient ouvrir la voie afin de :

1. Énumérer les types de graphes et d'arbres dont on peut calculer les supports en utilisant des règles de dérivations de support.

2. Extraire des règles d'associations pour les arbres et les graphes. Ce genre de règles seraient extrêmement intéressantes puisqu'elles auraient une application directe dans des domaines très dynamiques actuellement comme la bioinformatique ou la fouille de réseaux sociaux (par exemple, on pourrait avoir des règles stipulant qu'un graphe représentant X personnes et leurs relations sociales impliquerait un autre graphe contenant d'autre personnes inconnues et leurs relations !).

De plus, nous pensons que l'utilisation des classes d'équivalences permettrait de s'attaquer au problème inverse de l'extraction de motifs séquentiels qui consiste à générer des bases de données selon un ensemble de contraintes de supports. A notre connaissance, aucune approche n'a essayé de résoudre cette problématique dans le cadre des motifs séquentiels (alors qu'il existe plusieurs méthodes dans le cadre des itemsets [Mie03, WW05]).

Bibliographie

[AAB+05] M. H. Ali, W. G. Aref, R. Bose, A. K. Elmagarmid, A. Helal, I. Kamel et M. F. Mokbel : Nile-pdt : a phenomenon detection and tracking framework for data stream management systems. *In VLDB '05 : Proceedings of the 31st international conference on Very large data bases*, pages 1295–1298. VLDB Endowment, 2005.

[ABB+03] A. Arasu, B. Babcock, S. Babu, M. Datar, K. Ito, I. Nishizawa, J. Rosenstein et J. Widom : Stream : the stanford stream data manager (demonstration description). *In SIGMOD'03 : Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, pages 665–665, New York, NY, USA, 2003. ACM.

[ACc+03] D. Abadi, D. Carney, U. Çetintemel, M. Cherniack, C. Convey, C. Erwin, E. Galvez, M. Hatoun, A. Maskey, A. Rasin, A. Singer, M. Stonebraker, N. Tatbul, Y. Xing, R. Yan et S. Zdonik : Aurora : a data stream management system. *In SIGMOD '03 : Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, pages 666–666, New York, NY, USA, 2003. ACM.

[AFGY02] J. Ayres, J. Flannick, J. Gehrke et T. Yiu : Sequential pattern mining using bitmap representation. *In Proceedings of the 8th SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 02)*, pages 439–435, Alberta, Canada, 2002.

[Agg06] C. Aggarwal : On biased reservoir sampling in the presence of stream evolution. *In* Umeshwar Dayal, Kyu-Young Whang, David B. Lomet, Gustavo Alonso, Guy M. Lohman, Martin L. Kersten, Sang Kyun Cha et Young-Kuk Kim, éditeurs : *VLDB*, pages 607–618. ACM, 2006.

[Agg07] C. Aggarwal, éditeur. *Data Streams : Models and Algorithms*. Springer, 2007.

[AIS93] R. Agrawal, T. Imielinski et A. Swami : Mining association rules between sets of items in large database. *In Proceedings of the International Conference on Management of Data (ACM SIGMOD 93)*, pages 207–216, 1993.

[AS94] R. Agrawal et R. Srikant : Fast algorithms for mining association rules. *In* Jorge B. Bocca, Matthias Jarke et Carlo Zaniolo, éditeurs : *Proc. 20th Int. Conf. Very Large Data Bases, VLDB*, pages 487–499. Morgan Kaufmann, 12–15 1994.

[Atk89] K. Atkinson : *An Introduction to Numerical Analysis*. John Wiley & Sons, 2nd édition, 1989.

[Bar02] D. Barbara : Requirements for clustering data streams. *In SIGKDD Explor. Newsl.*, volume 3, pages 23–27, New York, NY, USA, 2002. ACM.

[Bay98] R. Bayardo : Efficiently mining long patterns from databases. *In SIGMOD '98 : Proceedings of the 1998 ACM SIGMOD international conference on Management of data*, pages 85–93, New York, NY, USA, 1998. ACM.

[BB00] J.F. Boulicaut et A. Bykowski : Frequent closures as a concise representation for binary data mining. *In Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 62–73, 2000.

[BBR00] J.F. Boulicaut, A. Bykowski et C. Rigotti : Approximation of frequency queries by means of free-sets. *In* Djamel A. Zighed, Henryk Jan Komorowski et Jan M. Zytkow, éditeurs : *PKDD*, volume 1910 de *Lecture Notes in Computer Science*, pages 75–85. Springer, 2000.

[BBR03] J.F. Boulicaut, A. Bykowski et C. Rigotti : Free-sets : A condensed representation of boolean data for the approximation of frequency queries. *Data Min. Knowl. Discov.*, 7(1):5–22, 2003.

[BCF+05] D. Burdick, M. Calimlim, J. Flannick, J. Gehrke et T. Yiu : Mafia : A maximal frequent itemset algorithm. *IEEE Transactions on Knowledge and Data Engineering*, 17(11):1490–1504, 2005.

[BCG03] J. Baixeries et G. Casas-Garriga : Sampling strategies for finding frequent sets. *In* Mohand-Said Hacid, Yves Kodratoff et Danielle Boulanger, éditeurs : *EGC*, volume 17 de *Revue des Sciences et Technologies de l'Information - série RIA ECA*, pages 159–170. Hermes Science Publications, 2003.

[BCG07] J.L. Balcázar et G. Casas-Garriga : Horn axiomatizations for sequential data. *Theor. Comput. Sci.*, 371(3):247–264, 2007.

[BDM02] B. Babcock, M. Datar et R. Motwani : Sampling from a moving window over streaming data, 2002.

[BR01] A. Bykowski et C. Rigotti : A condensed representation to find frequent patterns. *In Symposium on Principles of Database Systems*, 2001.

[BR03] A. Bykowski et C. Rigotti : Dbc : a condensed representation of frequent patterns for efficient mining. *Inf. Syst.*, 28(8):949–977, 2003.

[BTP+00] Y. Bastide, R. Taouil, N. Pasquier, G. Stumme et L. Lakhal : Mining frequent patterns with counting inference. *SIGKDD Explorations*, 2(2):66–75, 2000.

[Cal03] T. Calders : *Axiomatization and Deduction Rules for the Frequency of Itemsets.* Thèse de doctorat, University of Antwerp, Belgium, 2003.

[Cal08] T. Calders : Itemset frequency satisfiability : Complexity and axiomatization. *Accepted November 2007 for publication in Theoretical computer Science, to appear*, 2008.

[CDH+02] Y. Chen, G. Dong, J. Han, B. Wah et J. Wang : Multi-dimensional regression analysis of time-series data streams. *In Proceedings of the 28th International Conference on Very Large Databases (VLDB 02)*, pages 322–334, Hong Kong, China, 2002.

[CFPR00] C. Cortes, K. Fisher, D. Pregibon et A. Rogers : Hancock : a language for extracting signatures from data streams. *In Knowledge Discovery and Data Mining*, pages 9–17, 2000.

[CG02a] T. Calders et B. Goethals : Mining all non-derivable frequent itemsets. *In PKDD*, pages 74–85. Springer, 2002.

[CG02b] T. Calders et B. 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.

[CG03] T. Calders et B. Goethals : Minimal k-free representations of frequent sets. *In PKDD proceedings*, pages 71–82, 2003.

[CG05a] T. Calders et B. Goethals : Depth-first non-derivable itemset mining. *In SDM*, 2005.

[CG05b] T. Calders et B. Goethals : Quick inclusion-exclusion. *In Proceedings ECML-PKDD 2005 Workshop Knowledge Discovery in Inductive Databases*, volume 3933 de *LNCS*. Springer, 2005.

[CG06] G. Casas-Garriga : *Formal methods for mining structured objects.* Thèse de doctorat, Universitat Politècnica de Catalunya, 2006.

[CG07] T. Calders et B. Goethals : Non-derivable itemset mining. *DMKD*, 14(1):171–206, February 2007.

[Che52] H. Chernoff : A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations. *Annals of Mathematical Statistics*, 23:493–507, 1952.

[CHHP05] S. Cong, J. Han, J. Hoeflinger et D.A. Padua : A sampling-based framework for parallel data mining. *In* Keshav Pingali, Katherine A. Yelick et Andrew S. Grimshaw, éditeurs : *PPOPP*, pages 255–265. ACM, 2005.

[Chv83] V. Chvatal : *Linear Programming.* A Series of Books in the Mathematical Sciences. W. H. Freeman and Company, New York, 1983.

[CL03] J.H. Chang et W.S. Lee : Finding recent frequent itemsets adaptively over online data streams. *In KDD '03 : Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 487–492, New York, NY, USA, 2003. ACM.

[CRB06] T. Calders, C. Rigotti et J.F. Boulicaut : A survey on condensed representations for frequent sets. *Constraint Based Mining, Springer-Verlag, LNAI*, 3848:64–80, 2006.

[CWYM04] Y. Chi, H. Wang, P.S. Yu et R.R. Muntz : Moment : Maintaining closed frequent itemsets over a stream sliding window. *In Proceedings of the 4th IEEE International Conference on Data Mining (ICDM 04)*, pages 59–66, Brighton, UK, 2004.

[CWZ05] G. Chen, X. Wu et X. Zhu : Mining sequential patterns accross data streams. Rapport technique CS-05-04, University of Vermont, march 2005.

[DH00] P. Domingos et G. Hulten : Mining high-speed data streams. *In Knowledge Discovery and Data Mining*, pages 71–80, 2000.

[EHZ05] M. El-Hajj et O.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.

[GHP+03] G. Giannella, J. Han, J. Pei, X. Yan et P. Yu : Mining frequent patterns in data streams at multiple time granularities. *In In H. Kargupta, A. Joshi, K. Sivakumar and Y. Yesha (Eds.), Next Generation Data Mining, MIT Press*, 2003.

[Gib01] P.B. Gibbons : Distinct sampling for highly-accurate answers to distinct values queries and event reports. *In The VLDB Journal*, pages 541–550, 2001.

[GKMS01] A. Gilbert, Y. Kotidis, S. Muthukrishnan et M. Strauss : Quicksand : Quick summary and analysis of network data. DIMACS Technical Report, 2001.

[GLH06] R. Gemulla, W. Lehner et P.J. Haas : A dip in the reservoir : maintaining sample synopses of evolving datasets. *In VLDB '06 : Proceedings of the 32nd international conference on Very large data bases*, pages 595–606. VLDB Endowment, 2006.

[Goe02] B. Goethals : *Efficient Frequent Pattern Mining.* Thèse de doctorat, transnationale Universiteit Limburg, 2002.

[GVL96] G.H. Golub et C.F. Van Loan : *Matrix computations.* Johns Hopkins University Press, 3rd édition, 1996.

[GW97] B. Ganter et R. Wille : *Applied Lattice Theory : Formal Concept Analysis.* 1997.

[GZ01] K. Gouda et M.J. Zaki : Efficiently mining maximal frequent itemsets. *In ICDM*, pages 163–170, 2001.

[GZ05] K. Gouda et M.J. Zaki : Genmax : An efficient algorithm for mining maximal frequent itemsets. *Data Min. Knowl. Discov.*, 11(3):223–242, 2005.

[HCD+05] J. Han, Y. Chen, G. Dong, J. Pei, B.W. Wah, J. Wang et Y.D. Cai : Stream cube : An architecture for multi-dimensional analysis of data streams. *Distributed and Parallel Databases*, 18(2):173–197, 2005.

[HLC01] J. Hsu, C. Liu et A.L.P. Chen : Discovering nontrivial repeating patterns in music data. *IEEE Transactions on Multimedia*, 3(3):311–325, 2001.

[Hoe63] W. Hoeffding : Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, mars 1963.

[HPMa+00] J. Han, J. Pei, B. Mortazavi-asl, Q. Chen, U. Dayal et M. Hsu : Freespan : Frequent pattern-projected sequential pattern mining. *In Proceedings of the 6th International Conference on Knowledge Discovery and Data Mining (KDD 00)*, pages 355–359, Boston, USA, 2000.

[IR90] K. Ireland et M. Rosen : *A Classical Introduction to Modern Number Theory*. Springer-Verlag, 1990.

[JD88] A.K Jain et R.C. Dubes : *Algorithms for clustering data*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1988.

[KG02a] M. Kryszkiewicz et M. Gajek : Concise representation of frequent patterns based on generalized disjunction-free generators. *In* Ming-Shan Cheng, Philip S. Yu et Bing Liu, éditeurs : *PAKDD*, volume 2336 de *Lecture Notes in Computer Science*, pages 159–171. Springer, 2002.

[KG02b] M. Kryszkiewicz et M. Gajek : Why to apply generalized disjunction-free generators representation of frequent patterns ? *In ISMIS '02 : Proceedings of the 13th International Symposium on Foundations of Intelligent Systems*, pages 383–392, London, UK, 2002. Springer-Verlag.

[KPWD03] H.-C. Kum, J. Pei, W. Wang et D. Duncan : Approxmap : Approximate mining of consensus sequential patterns. *In Proceedings of the 3rd SIAM International Conference on Data Mining (ICDM 03)*, pages 311–315, San Francisco, CA, 2003.

[LC05] C. Luo et S.M. Chung : Efficient mining of maximal sequential patterns using multiple samples. *In SDM*, 2005.

[LLS04] H.-F. Li, S.Y. Lee et M.-K. Shan : An efficient algorithm for mining frequent itemsets over the entire history of data streams. *In Proceedings of the 1st International Workshop on Knowledge Discovery in Data Streams*, Pisa, Italy, 2004.

[LT91] M. Ledoux et M. Talagrand : *Probability in Banach spaces*. Springer, 1991.

[LZO99] N Lesh, M.J. Zaki et M. Ogihara : Mining features for sequence classification. *In* S. Chaudhuri et D. Madigan, éditeurs : *Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Diego, 1999. ACM Press.

[MCP98] F. Masseglia, F. Cathala et P. Poncelet : The PSP approach for mining sequential patterns. *In Proceedings of the 2nd European Symposium on Principles of Data Mining and Knowledge Discovery (PKDD 98)*, pages 176–184, Nantes, France, 1998.

[Mie03] T. Mielikäinen : On inverse frequent set mining. *In* Wenliang Du et Christopher W. Clifton, éditeurs : *Proceedings of the 2nd Workshop on Privacy Preserving Data Mining (PPDM), November 19, 2003, Melbourne, Florida, USA*, pages 18–23. IEEE Computer Society, 2003.

[MM02] G. Manku et R. Motwani : Approximate frequency counts over data streams. *In Proceedings of the 28th International Conference on Very Large Data Bases (VLDB 02)*, pages 346–357, Hong Kong, China, 2002.

[MM06] A. Marascu et F. Masseglia : Extraction de motifs séquentiels dans les flots de données d'usage du web. *In EGC*, pages 627–638, 2006.

[MT96] H. Mannila et H. Toivonen : Multiple uses of frequent sets and condensed representations (extended abstract). *In Knowledge Discovery and Data Mining*, pages 189–194, 1996.

[MT97] H. Mannila et H. Toivonen : Levelwise search and borders of theories in knowledge discovery. *Data Mining and Knowledge Discovery*, 1(3):241–258, 1997.

[MTV97] H. Mannila, H. Toivonen et A.I. Verkamo : Discovery of frequent episodes in event sequences. *Data Mining and Knowledge Discovery*, 1(3):259–289, 1997.

[Mut03] S. Muthukrishnan : Data streams : algorithms and applications, 2003.

[NJ07] A. Nandi et H. V. Jagadish : Effective phrase prediction. *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 219–230. ACM, 2007.

[Pap91] A. Papoulis : *Probability, Random Variables and Stochastic Processes*. 1991.

[Pap94] C. M. Papadimitriou : *Computational complexity*. Addison-Wesley, Reading, Massachusetts, 1994.

[PBTL99] N. Pasquier, Y. Bastide, R. Taouil et L. Lakhal : Discovering frequent closed itemsets for association rules. *Lecture Notes in Computer Science*, 1540:398–416, 1999.

[PCL+05] M. Plantevit, Y.W. Choong, A. Laurent, D. Laurent et M. Teisseire : M^2SP : Mining sequential patterns among several dimensions. *In PKDD*, pages 205–216, 2005.

[PHMa+01] J. Pei, J. Han, B. Mortazavi-asl, H. Pinto, Q. Chen et U. Dayal : Prefixspan : Mining sequential patterns efficiently by prefix-projected pattern growth. *In Proceedings of 17th International Conference on Data Engineering (ICDE 01)*, pages 215–224, Heidelberg, Germany, 2001.

[PHMa+04] J. Pei, J. Han, B. Mortazavi-asl, J. Wang, H. Pinto, Q. Chen, U. Dayal et M. Hsu : Mining sequential patterns by pattern-growth : The prefixspan approach. *IEEE Trans. Knowl. Data Eng.*, 16(11):1424–1440, 2004.

[PHMaZ00] J. Pei, J. Han, B. Mortazavi-asl et H. Zhu : Mining access patterns efficiently from web logs. *In PAKDD*, pages 396–407, 2000.

[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, 2001.

[PHW02] J. Pei, J. Han et W. Wang : Mining sequential patterns with constraints in large databases. *In Proceedings of the 10th International Conference on Information and Knowledge Management (CIKM 02)*, pages 18–25, MCLean, USA, 2002.

[PK03] K. Park et M. Kanehisa : Prediction of protein subcellular locations by support vector machines using compositions of amino acids and amino acid pairs. *Bioinformatics*, 19(13):1656–1663, 2003.

[PZOD99] S. Parthasarathy, M.J. Zaki, M. Ogihara et S. Dwarkadas : Incremental and interactive sequence mining. *In 8th International Conference on Information and Knowledge Management*, pages pp 251–258, 1999.

[RPT06] C. Raissi, P. Poncelet et M. Teisseire : Speed : Mining maximal sequential patterns over data streams. *In Proceedings of the 3rd IEEE International Conference On Intelligent Systems (IS2006)*, Westminster, UK, 2006.

[SA96] R. Srikant et R. Agrawal : Mining sequential patterns : Generalizations and performance improvements. *In Proceedings of the 5th International Conference on Extending Database Technology (EDBT 96)*, pages 3–17, Avignon, France, 1996.

[SCDT00] J. Srivastava, R. Cooley, M. Deshpande et P. Tan : Web usage mining : Discovery and applications of usage patterns from web data. *SIGKDD Explorations*, 1:12–23, 2000.

[SCR04] A. Soulet, B. Crémilleux et F. Rioult : Condensed representation of emerging patterns. *In* Honghua Dai, Ramakrishnan Srikant et Chengqi Zhang, éditeurs : *PAKDD*, volume 3056 de *Lecture Notes in Computer Science*, pages 127–132. Springer, 2004.

[SD05] A. Satyanarayana et I. Davidson : A dynamic adaptive sampling algorithm (dasa) for real world applications : Finger print recognition and face recognition. *In* Springer, éditeur : *Foundations of Intelligent Systems*, volume 3488, pages 631–640, 2005.

[Sri95] R. Agrawal R. Srikant : Mining sequential patterns. *In Proceedings of the 11th International Conference on Data Engineering (ICDE 95)*, pages 3–14, Tapei, Taiwan, 1995.

[TCY03] W.-G. Teng, M.-S. Chen et P.S. Yu : A regression-based temporal patterns mining schema for data streams. *In Proceedings of the 29th International Conference on Very Large Data Bases (VLDB 03)*, pages 93–104, Berlin, Germany, 2003.

[Toi96] H. Toivonen : Sampling large databases for association rules. *In* T. M. Vijayaraman, Alejandro P. Buchmann, C. Mohan et Nandlal L. Sarda, éditeurs : *VLDB*, pages 134–145. Morgan Kaufmann, 1996.

[TYH05] P. Tzvetkov, X. Yan et J. Han : Tsp : Mining top-k closed sequential patterns. *Knowl. Inf. Syst.*, 7(4):438–457, 2005.

[Vit85] J. Vitter : Random sampling with a reservoir. *ACM Trans. Math. Softw.*, 11(1):37–57, 1985.

[WH04] J. Wang et J. Han : Bide : Efficient mining of frequent closed sequences. *In* Daniel Barbará et Chandrika Kamath, éditeurs : *ICDE*, pages 79–90. IEEE Computer Society, 2004.

[WK91] S.M. Weiss et C.A. Kulikowski : *Computer Systems That Learn. Classification and Prediction Methods from Statistics, Neural Nets, Machine Learning, and Expert Systems*. Morgan Kaufmann, San Francisco, 1991.

[WM03] T. Washio et H. Motoda : State of the art of graph-based data mining. *SIGKDD Explor. Newsl.*, 5(1):59–68, 2003.

[WW05] Y. Wang et X. Wu : Approximate inverse frequent itemset mining : Privacy, complexity, and approximation. *In ICDM '05 : Proceedings of the Fifth IEEE International Conference on Data Mining*, pages 482–489, Washington, DC, USA, 2005. IEEE Computer Society.

[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.

[YCLZ04] J. Xu Yu, Z. Chong, H. Lu et A. Zhou : False positive or false negative : mining frequent itemsets from high speed transactional data streams. *In VLDB '04 : Proceedings of the Thirtieth international conference on Very large data bases*, pages 204–215. VLDB Endowment, 2004.

[YHA03] X. Yan, J. Han et R. Afshar : Clospan : Mining closed sequential patterns in large databases. *In* Daniel Barbará et Chandrika Kamath, éditeurs : *SDM*. SIAM, 2003.

[YKW06] Z. Yang, M. Kitsuregawa et Y. Wang : Paid : Mining sequential patterns by passed item deduction in large databases. *In IDEAS*, pages 113–120. IEEE Computer Society, 2006.

[YWK05] Z. Yang, Y. Wang et M. Kitsuregawa : Lapin : Effective sequential pattern mining algorithms by last position induction. Rapport technique, Tokyo University, 2005.

[YWYH02] J. Yang, W. Wang, P.S. Yu et J. 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] M.J. Zaki : Spade : An efficient algorithm for mining frequent sequences. *Machine Learning*, 42(1/2):31–60, 2001.

[ZH02] M.J. Zaki et C. 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.

[Zhu] Y. Zhu : Fast approaches to simple problems in financial time series streams.

[ZXY08] G. Dong Z. Xing, J. Pei et P. S. Yu : Mining sequence classifiers for early prediction. *In Proceedings of the 2008 SIAM International Conference on Data Mining (SDM'08)*, Atlanta, GA, April 24-26 2008.

Articles publiés dans le cadre de cette thèse

Revues internationales avec comité de lecture :

- *Mining Conjunctive Sequential Patterns*, C. Raïssi, T. Calders and P. Poncelet *Data Mining and Knowledge Discovery, Volume 17, August 2008, To appear.*

- *Towards a new approach for mining frequent itemsets on data stream*, C. Raïssi, P. Poncelet and M. Teisseire *Journal of Intelligent Information Systems, Volume 28, February 2007, pp. 23-36.*

Revues nationales avec comité de lecture :

- *Extraction de motifs pour les flots de données*, C. Raïssi *Revue I3, Numéro Hors série 2007, Cépadués Edition, July 2007, pp. 47-73.*

- *Random Sampling over Data Streams for Sequential Pattern Mining*, C. Raïssi and P. Poncelet *La Revue Modulad, Numéro 36, May 2007, pp. 61-66 (Special Issue on Data Stream Analysis).*

Conférences Internationales avec comité de lecture :

- *Mining Conjunctive Sequential Patterns*, C. Raïssi, T. Calders and P. Poncelet *In Proc. of Principles and Practice of Knowledge Discovery in Databases 2008 (PKDD 2008), Antwerp, Belgium. September 2008, To appear.*

- *Mining Multidimensional Sequential Patterns over Data Streams*, C. Raïssi and M. Plantevit. *In Proc. of 2008 International Conference on Data Warehousing and Knowledge Discovery (DAWAK'08), Torino, Italia. September 2008, To appear.*

– *Sampling For Sequential Pattern Mining : From Static Databases to Data Streams*, C. Raïssi and P. Poncelet. *In Proc. of 2007 International Conference on Data Mining (IEEE ICDM 2007), Omaha NE, USA. October 2007, pp. 631-636.*

– *SPEED : Mining Maximal Sequential Patterns over Data Streams*, C. Raïssi, P. Poncelet and M. Teisseire. *In Proc. of 3rd Conference on Intelligent Systems (IEEE IS2006), London, United-Kingdom. September 2006.*

Conférences Nationales avec comité de lecture :

– *Vers une nouvelle approche d'extraction des motifs séquentiels non-dérivables*, C. Raïssi and P. Poncelet. *In Proc. of Extraction et Gestion des Connaissances (EGC'2007), Namur, Belgium. January 2007, pp. 307-318.*

– *Need For Speed : Mining Sequential Patterns in Data Streams*, C. Raïssi and P. Poncelet. *In Proc. of Base de Données Avancées Conference (BDA'2005), St-Malo, France. October 2005, pp. 151-162.*

– *Annotation automatique de documents*, L. Abrouk, A. Gouaich and C. Raïssi *In Proc. of Méthodes et Outils pour la Conception et la Réalisation des Systèmes d'Information (INFORSID), Hammamet, Tunisia. June 2006.*

Ateliers avec comité de lecture :

– *Web Analysing Traffic Challenge : Description and Results*, C. Raïssi, J. Brissaud, G. Dray, P. Poncelet, M. Roche and M. Teisseire *In Proc. of the Discovery Challenge ECML/PKDD 2007, Warsaw, Poland. September 2007.*

– *Random Sampling over Data Streams for Sequential Pattern Mining*, C. Raïssi *European Workshop on Data Stream Analysis (WSDA 2007), Caserta, Italy, March 2007, pp. 61-66.*

– *Extraction de motifs séquentiels dans des data streams*, C. Raïssi and P. Poncelet *GDR I3 - Groupe 3.4 - "Fouille de Données", Paris, France. May 2005.*

– *Détection d'intrusions : de l'utilisation de signatures statistiques*, P. Gupta, C. Raïssi, G. Dray, P. Poncelet and J. Brissaud *EGC'2008* 5^{th} *Workshop on Complex Data Mining. Nice,*

France. January 2008.

– *FIDS : Extraction efficace d'itemsets dans des flots de données*, C. Raïssi *EGC'2006 Workshop on temporal data mining. (EGC2006), Lille, France. January 2006.*

Annexes

Annexe A

Quelques règles de dérivations

Règles pour la séquence $\langle(a)(b)\rangle$:

$$
\begin{array}{lrcl}
(1) & \langle(a)(b)\rangle & \geqslant & 0 \\
(2) & \langle(b)\rangle & \geqslant & \langle(a)(b)\rangle \\
(3) & \langle(a)\rangle & \geqslant & \langle(a)(b)\rangle \\
(4) & \langle()\rangle & \geqslant & \langle(b)\rangle \\
(5) & \langle()\rangle & \geqslant & \langle(a)\rangle
\end{array}
$$

Règles pour la séquence $\langle(a)(b)(c)\rangle$:

$$
\begin{array}{lrcl}
(1) & \langle(a)(b)(c)\rangle & \geqslant & 0 \\
(2) & \langle(b)(c)\rangle & \geqslant & \langle(a)(b)(c)\rangle \\
(3) & \langle(a)(b)\rangle & \geqslant & \langle(a)(b)(c)\rangle \\
(4) & \langle(a)(c)\rangle & \geqslant & \langle(a)(b)(c)\rangle \\
(5) & \langle(c)\rangle & \geqslant & \langle(b)(c)\rangle \\
(6) & \langle(b)\rangle & \geqslant & \langle(b)(c)\rangle \\
(7) & \langle(a)\rangle & \geqslant & \langle(a)(c)\rangle \\
(8) & \langle(a)\rangle & \geqslant & \langle(a)(b)\rangle \\
(9) & \langle(c)\rangle & \geqslant & \langle(a)(c)\rangle \\
(10) & \langle(b)\rangle & \geqslant & \langle(a)(b)\rangle \\
(11) & \langle()\rangle & \geqslant & \langle(c)\rangle \\
(12) & \langle()\rangle & \geqslant & \langle(b)\rangle \\
(13) & \langle()\rangle & \geqslant & \langle(a)\rangle \\
(14) & \langle()\rangle + \langle(a)(b)\rangle + \langle(b)(c)\rangle & \geqslant & \langle(a)(b)(c)\rangle + \langle(a)(c)\rangle + \langle(b)\rangle
\end{array}
$$

Règles pour la séquence $\langle(a)(b)(b)\rangle$:

$$
\begin{array}{lrcl}
(1) & \langle(a)(b)(b)\rangle & \geqslant & 0 \\
(2) & \langle(a)(b)\rangle & \geqslant & \langle(a)(b)(b)\rangle \\
(3) & \langle(b)(b)\rangle & \geqslant & \langle(a)(b)(b)\rangle \\
(4) & \langle(b)\rangle & \geqslant & \langle(a)(b)\rangle \\
(5) & \langle(b)\rangle & \geqslant & \langle(b)(b)\rangle \\
(6) & \langle(a)\rangle & \geqslant & \langle(a)(b)\rangle \\
(7) & \langle()\rangle & \geqslant & \langle(a)\rangle \\
(8) & \langle()\rangle & \geqslant & \langle(b)\rangle
\end{array}
$$

Règles pour la séquence $\langle(a)(b)(a)\rangle$:

$$
\begin{array}{lrcl}
(1) & \langle(a)(b)(a)\rangle & \geqslant & 0 \\
(2) & \langle(b)(a)\rangle & \geqslant & \langle(a)(b)(a)\rangle \\
(3) & \langle(a)(b)\rangle & \geqslant & \langle(a)(b)(a)\rangle \\
(4) & \langle(a)(a)\rangle & \geqslant & \langle(a)(b)(a)\rangle \\
(5) & \langle(b)\rangle & \geqslant & \langle(b)(a)\rangle \\
(6) & \langle(a)\rangle & \geqslant & \langle(b)(a)\rangle \\
(7) & \langle(a)\rangle & \geqslant & \langle(a)(a)\rangle \\
(8) & \langle(a)\rangle & \geqslant & \langle(a)(b)\rangle \\
(9) & \langle(b)\rangle & \geqslant & \langle(a)(b)\rangle \\
(10) & \langle()\rangle & \geqslant & \langle(b)\rangle \\
(11) & \langle()\rangle & \geqslant & \langle(a)\rangle \\
(12) & \langle()\rangle + \langle(a)(b)\rangle + \langle(b)(a)\rangle & \geqslant & \langle(a)\rangle + \langle(a)(b)(a)\rangle + \langle(b)\rangle
\end{array}
$$

Extraction de séquences fréquentes : des bases de données statiques aux flots de données

Il est reconnu aujourd'hui que l'être humain est généralement noyé sous une profusion d'informations et que sa capacité d'analyse n'est plus capable de faire face au volume sans cesse croissant de données. C'est dans ce contexte qu'est né le processus d'Extraction de Connaissance dans les bases de Données. Un des buts de ce processus est de passer d'un grand volume d'informations à un petit ensemble de connaissances à fortes valeurs ajoutées pour l'analyste ou le décideur. De plus, le processus d'ECD n'est pas un processus monolithique et univoque au cours duquel il s'agirait d'appliquer un principe général à tous les types de données stockées ou récupérées. Ainsi, une des étapes de ce processus qu'est la fouille de données peut se dériver sous plusieurs formes tels que : le clustering, la classification, l'extraction d'itemset et de règles d'associations, l'extraction de structures plus complexes tels que les épisodes, les graphes ou comme dans le cadre de cette thèse l'extraction de motifs séquentiels.

Malheureusement, dans un monde sans cesse en évolution, le contexte dans lequel les travaux d'ECD ont été définis ces dernières années considérait que les données, sur lesquelles la fouille était réalisée, étaient disponibles dans des bases de données statiques. Aujourd'hui, suite au développement de nouvelles technologies et applications associées, nous devons faire face à de nouveaux modèles dans lesquels les données sont disponibles sous la forme de flots. Une question se pose alors : *quid des approches d'extraction de connaissances traditionnelles ?*

Dans ce mémoire, nous présentons un ensemble de résultat sur les motifs séquentiels dans les bases de données d'un point de vue des représentations condensées et des méthodes d'échantillonnage puis nous étendons nos différentes approches afin de prendre en compte le nouveau modèle des flots de données. Nous présentons des algorithmes permettant ainsi l'extraction de motifs séquentiels (classiques et multidimensionnels) sur les flots. Des expérimentations menées sur des données synthétiques et sur des données réelles sont rapportées et montrent l'intérêt de nos propositions.

Mots-clés : Extraction de connaissances, fouille de données, motifs séquentiels, séquences fréquentes, base de données de séquences, représentations condensées, flots de données, échantillonnage, règles d'association.

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