Conférence Francophone sur l'Apprentissage Automatique (CAp 2013), Lille, France, Juillet 2013

Approximation de bordures de motifs fréquents par le calcul de traverses minimales approchées d'hypergraphes

Nicolas Durand et Mohamed Quafafou

Aix-Marseille Université - LSIS UMR 7296
Avenue Escadrille Normandie Niemen - 13397 Marseille - France

Résumé

Cet article présente une nouvelle approche d'approximation de la bordure négative et de la bordure positive de motifs fréquents. Notre approche s'appuie sur le passage d'une bordure à l'autre par le calcul de traverses minimales d'hypergraphes. Nous proposons alors une nouvelle méthode de génération de traverses minimales approchées reposant sur la réduction d'hypergraphes. Les expérimentations sur différents jeux de données montrent que notre proposition pour approximer les bordures produit des résultats prometteurs.

Mots-clef : ensembles/motifs fréquents, bordure, hypergraphe, approximation.

1 Introduction

La découverte de motifs fréquents a été initiée par Agrawal et al. [AIS93]. Elle est rapidement devenue une tâche importante de la fouille de données. Elle correspond à trouver les ensembles d'items (i.e. valeurs d'attributs) qui apparaissent simultanément dans au moins un certain nombre de transactions (i.e. objets) enregistrés dans une base de données. Depuis d'autres types de motifs ont vu le jour, comme par exemple les motifs fermés [PBTL99], les motifs libres [BBR00] et les motifs émergents [DL99]. L'usage premier des motifs est la création de règles d'association [AIS93]. Les usages se sont ensuite étendus à d'autres tâches de la fouille de données comme la classification supervisée [RBF05] et non-supervisée [DC02]. Deux points sont importants dans la découverte de motifs. Le premier est la réduction de l'espace de recherche étant donné le caractère exponentiel du calcul. Le second point est la réduction du nombre de motifs générés. Pour cela, des représentations condensées de motifs telles que les fermés fréquents ou les libres fréquents ont été proposées. A partir de ceux-ci, la collection complète des motifs fréquents peut être regénérée sans parcourir la base de données. Leur nombre est bien inférieur au nombre total de motifs fréquents. De plus, les propriétés de ces motifs particuliers permettent d'effectuer des élagages supplémentaires dans l'espace de recherche. Les motifs fréquents maximaux (au sens de l'inclusion) représentent une collection réduite de motifs. Ils correspondent à un sous-ensemble des motifs fermés fréquents. Bien que la regénération de tous les motifs fréquents soit possible à partir de ces motifs, ils ne sont pas considérés comme des représentations condensées de motifs. En effet, pour obtenir les valeurs de fréquence des motifs regénérés, il faut parcourir les données ce qui n'est pas le cas à partir des motifs fermés, par exemple. Les motifs fréquents maximaux et les motifs infréquents minimaux représentent respectivement la bordure positive et la bordure négative de l'ensemble des motifs fréquents [MT97]. Ces deux bordures sont liées et il est possible de passer d'une bordure à l'autre par le calcul de traverses minimales d'hypergraphes [MT97, DMP03].

Une autre approche pour réduire le nombre de motifs est l'approximation. Dans cette perspective, nous avons une double contribution. Nous présentons une nouvelle approche d'approximation de la bordure négative et de la bordure positive de motifs fréquents. Elle s'appuie sur le passage d'une bordure à l'autre par le calcul de traverses minimales d'hypergraphes. L'approximation est alors obtenue par le calcul de traverses minimales approchées. Notre deuxième contribution est une nouvelle méthode de calcul de traverses minimales approchées via la réduction de l'hypergraphe initial. Des expérimentations ont été réalisées sur des données de différents types afin d'évaluer le nombre de motifs produits et la distance entre les bordures approchées calculées et les bordures exactes. Nous avons aussi confronté nos résultats avec ceux obtenus en

substituant notre méthode de traverses minimales approchées par l'algorithme présenté dans [RZC10] qui génère des traverses minimales avec un certain nombre d'exceptions possibles.

L'article est organisé comme suit. La section 2 évoque les travaux liés à l'approximation de bordures et à l'approximation de traverses minimales d'hypergraphes. Dans la section 3, nous présentons les notations et les notions utilisées. Notre approche d'approximation de bordures est détaillée dans la section 4. Nous présentons ensuite, dans la section 5, notre méthode de calcul de traverses minimales approchées. Les expérimentations et les résultats sont présentés et discutés dans la section 6. Enfin, nous concluons et évoquons les perspectives de notre travail en section 7.

2 Travaux existants

L'approximation d'une collection de motifs fréquents par les k meilleurs ensembles couvrants a été étudié dans [AGM04]. En entrée de l'algorithme d'approximation, il est spécifié soit l'ensemble complet des motifs fréquents soit la bordure positive. Les auteurs ont expliqué les difficultés liées à l'utilisation d'un algorithme glouton d'approximation à partir de la bordure positive pour obtenir k ensembles couvrants faisant partie de la collection intitiale. L'approximation de bordures positives et de bordures négatives a été étudiée algorithmiquement dans [Bol07]. Dans ces travaux, le calcul d'approximations de bordure à partir des données s'effectue directement contrairement aux travaux précédents. L'auteur montre qu'il y a une indication forte qu'il n'y a pas d'algorithme d'approximation pour le calcul de la bordure positive avec une facteur d'approximation raisonnable. Par contre, le calcul de la bordure négative peut être approximé par un algorithme glouton en temps polynomial avec un facteur d'approximation de $\lceil ln(t-s) \rceil + 1$ où t est le nombre de transactions et s le seuil minimal de fréquence.

Notre approche calcule une bordure approchée à partir d'une bordure entièrement donnée en entrée. En cela, nous nous rapprochons des premiers travaux mentionnés dans cette section. Cependant, nous ne cherchons pas une couverture de la bordure initiale. Les motifs de la bordure approximative ne font pas nécessairement partie de la collection initiale. Notre but étant de générer une bordure postitive approchée, la bordure positive exacte est donc spécifiée en entrée. L'approximation est réalisée lors du calcul de traverses minimales utilisé pour le passage à la bordure négative qui est alors approchée. Le retour à la bordure positive produit au final une bordure positive approchée. A notre connaissance, c'est la première fois qu'une telle approche est proposée.

Le calcul de traverses minimales (aussi appelé "minimal hitting sets" en anglais) est un point central dans la théorie des hypergraphes [Ber89]. Les algorithmes de calcul de traverses minimales proviennent de différents domaines : les graphes [KS05], la logique [FK96, EG02] et la fouille de données [DL05, BMR03, HBC07]. C'est un problème NP-difficile. Les algorithmes d'appoximation de traverses minimales sont peu nombreux. De plus, il y a plusieurs façons d'appréhender l'approximation. Certains travaux se basent sur une mesure d'évaluation d'un coût pour former un ensemble de traverses minimales approximatives [AvG09]. Des travaux calculent approximativement des traverses minimales pour en obtenir quelques unes voire une seule [RS02]. Les travaux générant un véritable ensemble de traverses minimales approchées sont rares. Nous pouvons citer ceux basés sur une approche évolutionnaire [VØ00] où la transversalité et la minimalité sont transcrits dans une fonction objectif. Nous trouvons aussi des travaux où un certain nombre d'exceptions liées à la transversalité sont autorisées. L'algorithme présenté dans [RZC10], que nous appelons δ-$MTminer$ en référence à $MTminer$ [HBC07], produit des traverses minimales où chacune d'entre elles peut ne pas couper jusqu'à δ hyperarêtes de l'hypergraphe.

Nous avons une approche différente des précédentes pour calculer des traverses minimales approchées. Nous proposons d'appliquer une réduction de l'hypergraphe et ensuite de calculer les traverses minimales de l'hypergraphe réduit qui seront considérées approchées pour l'hypergraphe initial. Cette réduction s'inspire de travaux dans le domaine de la segmentation d'images [DBRL12] où un algorithme de réduction utilisant les intersections des hyperarêtes d'un hypergraphe est proposée. Nous n'avons gardé que le principe et proposons un algorithme de réduction spécialement conçu pour calculer ensuite des traverses minimales.

3 Préliminaires

Soit $\mathcal{D} = (\mathcal{T}, \mathcal{I}, \mathcal{R})$ un contexte de fouille de données, $\mathcal{T}$ un ensemble de transactions, $\mathcal{I}$ un ensemble d'items et $\mathcal{R} \subseteq \mathcal{T} \times \mathcal{I}$ une relation binaire entre les transactions et les items. Chaque couple $(t, i) \in \mathcal{R}$ signifie que la transaction t est en relation avec l'item i. La table 1 donne un exemple de contexte de fouille de données ayant 6 transactions et 8 items (notés $A \ldots H$). Un motif est un sous-ensemble de $\mathcal{I}$. Nous

utilisons une notation de type chaîne de caractères pour ces ensembles, par exemple AB pour $\{A, B\}$. Le complémentaire du motif X (par rapport à l'ensemble $\mathcal{I}$) est noté $\overline{X}$. Une transaction t supporte un motif X ssi $\forall i \in X, (t, i) \in \mathcal{R}$. Un motif X est fréquent si le nombre de transactions le supportant dépasse (ou est égal à) un seuil minimum fixé (noté *minsup*). L'ensemble des motifs fréquents est noté S. Sur l'exemple de la table 1, si *minsup*=3 alors le motif H est fréquent car 4 transactions le supportent (t_3, t_4, t_5 et t_6). AE n'est pas fréquent car seules t_1 et t_3 le supportent.

Id	Items
t_1	A C E G
t_2	B C E G
t_3	A C E H
t_4	A D F H
t_5	B C F H
t_6	B C E F H

TABLE 1 – Exemple de base de données transactionelle

La notion de bordure de motifs fréquents a été introduite dans [MT97] (cf. définition 1).

Définition 1 (Bordures positive et négative) *La bordure positive (resp. négative) de S, notée $Bd^+(S)$ (resp. $Bd^-(S)$), est constituée par les motifs fréquents maximaux (resp. infréquents minimaux) (au sens de l'inclusion) de $\mathcal{D}$.*
$Bd^+(S) = \{X \in S \mid \forall Y \; tq \; X \subset Y, \; Y \notin S\}$
$Bd^-(S) = \{X \in 2^{\mathcal{I}} \setminus S \mid \forall Y \; tq \; Y \subset X, \; Y \in S\}$

Sur notre exemple avec *minsup*=3, $Bd^+(S) = \{A, BC, CE, CH, FH\}$ et $Bd^-(S) = \{D, G, AB, AC, AE, AF, AH, BE, BF, BH, CF, EF, EH\}$.

Avant de présenter les relations entre les bordures positive et négative de motifs fréquents, nous introduisons les notions d'hypergraphe (cf. définition 2) et de traverses minimales d'un hypergraphe (cf. définition 3).

Définition 2 (Hypergraphe) *Un hypergraphe $\mathcal{H} = (V, E)$ est constitué d'un ensemble V de sommets et d'un ensemble E d'hyperarêtes. Chaque hyperarête $e \in E$ est un ensemble de sommets inclus ou égal à V.*

Définition 3 (Traverse et traverse minimale) *Soit $\mathcal{H}$ un hypergraphe, l'ensemble de ses traverses est : $Tr(\mathcal{H}) = \{\tau \subseteq V \mid \forall e_i \in E, \tau \cap e_i \neq \emptyset\}$. Une traverse τ de $\mathcal{H}$ est dite minimale ssi aucun de ses sous-ensembles n'est une traverse de $\mathcal{H}$. L'ensemble des traverses minimales de $\mathcal{H}$ est noté $MinTr(\mathcal{H})$.*

Le passage de la bordure positive à la bordure négative, et le passage de la bordure négative à la bordure positive, sont spécifiés dans les propriétés 1 et 2.

Propriété 1 *[MT97]*
$Bd^-(S) = MinTr(\overline{Bd^+(S)})$ où $\overline{Bd^+(S)}$ représente l'hypergraphe dont les sommets sont les items de $\mathcal{I}$ et les hyperarêtes sont les complémentaires des motifs de la bordure positive de S.

Propriété 2 *[DMP03]*
$Bd^+(S) = \overline{MinTr(Bd^-(S))}$ où $Bd^-(S)$ représente l'hypergraphe dont les sommets sont les items de $\mathcal{I}$ et les hyperarêtes sont les motifs de la bordure négative de S.

Le terme *dualisation* désigne l'utilisation des propriétés précédentes pour déterminer la bordure positive à partir de la bordure négative, et inversement.

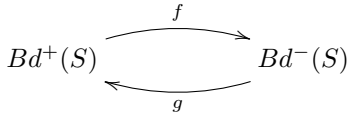
4 Approche proposée d'approximation de bordures

L'approche que nous proposons pour approcher les bordures exploite les dualisations entre la bordure positive et la bordure négative.

Soient f et g les fonctions qui permettent respectivement de passer de la bordure positive à la bordure négative et de la bordure négative à la bordure positive :

$$f : \left| \begin{array}{ccc} 2^{\mathcal{I}} & \rightarrow & 2^{\mathcal{I}} \\ x & \mapsto & MinTr(\overline{x}) \end{array} \right. \qquad g : \left| \begin{array}{ccc} 2^{\mathcal{I}} & \rightarrow & 2^{\mathcal{I}} \\ x & \mapsto & \overline{MinTr(x)} \end{array} \right.$$

Le schéma suivant permet de visualiser le passage d'une bordure à l'autre :



Le principe de notre approche est de remplacer la fonction f par une fonction $\widetilde{f}$ effectuant un calcul approximatif de la bordure négative.

La nouvelle fonction $\widetilde{f}$ utilise un calcul de traverses minimales approchées noté $\widetilde{MinTr}$:

$$\widetilde{f} : \left| \begin{array}{ccc} 2^{\mathcal{I}} & \rightarrow & 2^{\mathcal{I}} \\ x & \mapsto & \widetilde{MinTr}(\overline{x}) \end{array} \right.$$

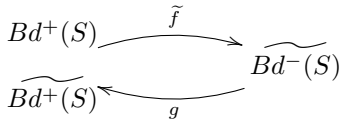
A partir d'une bordure positive, l'approche permet alors de calculer une bordure négative approximative notée $\widetilde{Bd^-(S)}$:

$$\widetilde{f}(Bd^+(S)) = \widetilde{MinTr}(\overline{Bd^+(S)}) = \widetilde{Bd^-(S)}$$

Le retour à la bordure positive (via la fonction g) permet au final d'obtenir une bordure positive approchée notée $\widetilde{Bd^+(S)}$:

$$g(\widetilde{Bd^-(S)}) = \overline{MinTr(\widetilde{Bd^-(S)})} = \widetilde{Bd^+(S)}$$

Nous pouvons visualiser cela de la façon suivante :



Notre approche produit donc la bordure négative approchée $\widetilde{Bd^-(S)}$ et la bordure positive approchée $\widetilde{Bd^+(S)}$ correspondante.

Reprenons notre exemple et calculons les bordures approchées :
$\widetilde{Bd^-(S)} = \widetilde{f}(Bd^+(S)) = \widetilde{MinTr}(\overline{Bd^+(S)})$
$= \widetilde{MinTr}(\{\overline{A}, \overline{BC}, \overline{CE}, \overline{CH}, \overline{FH}\}$
$= \widetilde{MinTr}(\{BCDEFGH, ADEFGH, ABDFGH, ABDEFG, ABCDEG\})$
Supposons que le calcul de traverses minimales approchées nous donne :
$\widetilde{Bd^-(S)} = \{D, E, G, AF, AH, BF, BH\}$.
Dualisons pour obtenir la bordure positive approchée :
$\widetilde{Bd^+(S)} = g(\widetilde{Bd^-(S)}) = \overline{MinTr(\widetilde{Bd^-(S)})}$
$= \{\overline{ABDEG}, \overline{DEFGH}\} = \{CFH, ABC\}$.

Nous remarquons que ABC n'est pas un motif fréquent ni existant dans les données, mais A, B, C et BC sont des motifs fréquents. CFH n'est pas fréquent (son support vaut 2) mais il l'est presque.

Notons que les hyperarêtes de l'hypergraphe formé par les complémentaires des motifs de la bordure positive, ont de fortes intersections. Sur notre exemple, nous avons les hyperarêtes : $\{BCDEFGH, ADEFGH, ABDFGH, ABDEFG, ABCDEG\}$. Cette observation est exploitée par le calcul de traverses minimales approchées que nous proposons dans la section suivante.

5 Calcul proposé de traverses minimales approchées

L'approche présentée dans la section précédente évoque un calcul de traverses minimales approchées d'hypergraphes. Afin de la compléter, nous proposons une méthode permettant de déterminer de telles traverses. Notre méthode repose sur la réduction de l'hypergraphe initial. Le but est alors de calculer des traverses minimales sur l'hypergraphe réduit (plus petit que l'hypergraphe initial). La réduction introduit la part d'approximation. L'algorithme proposé de réduction est spécialement conçu pour le calcul de traverses minimales et exploite la forte intersection des hyperarêtes de l'hypergraphe formé par les complémentaires des motifs de la bordure positive.

La méthode proposée est constituée de 2 phases :
1. réduction de l'hypergraphe
2. calcul (exact) de traverses minimales sur l'hypergraphe réduit.

Au final, les traverses minimales obtenues sur l'hypergraphe réduit sont déclarées comme les traverses minimales approchées de l'hypergraphe initial.

5.1 Réduction de l'hypergraphe

La réduction de l'hypergraphe initial est basée sur les intersections d'hyperarêtes et le nombre d'occurrences de chaque sommet. L'algorithme 1 présente cette réduction qui s'effectue en trois étapes :
1. calcul du nombre d'occurrences de chaque sommet dans les hyperarêtes de l'hypergraphe,
2. calcul des intersections de tous les couples d'hyperarêtes et construction d'un graphe valué,
3. sélection d'arêtes du graphe valué précédent et génération de l'hypergraphe réduit.

L'algorithme est en $O(m^2)$ où m est la cardinalité de l'ensemble des hyperarêtes de l'hypergraphe initial. Remarquons que nous pouvons généraliser notre algorithme à d'autres usages que les bordures en l'appliquant à chaque sous-hypergraphe connecté maximal ("composante connexe") de l'hypergraphe initial. Si les hyperarêtes sont disjointes, alors l'hypergraphe n'est pas réduit.

5.1.1 Etapes 1 et 2 : génération du graphe valué

Pour un hypergraphe $\mathcal{H} = (V, E)$ où $|V| = n$ et $|E| = m$, l'algorithme construit un graphe valué $G=(V', E')$ où $V' = \{v'_i\}$ avec $i = 1, \ldots, m$ et $E' = \{e'_k\}$ avec $k = 1, \ldots, l$. Un sommet v'_i représente une hyperarête e_i de $\mathcal{H}$. Nous notons $\psi : E \rightarrow V'$ la fonction bijective qui associe une hyperarête e_i à un sommet v'_i. Une arête entre v'_i et v'_j montre que l'intersection entre les hyperarêtes $\psi^{-1}(v'_i)$ et $\psi^{-1}(v'_j)$ (e_i et e_j dans $\mathcal{H}$) est non vide. Le poids d'une arête est basée sur le nombre d'occurrences de chaque sommet de l'intersection correspondante. Pour évaluer le poids d'une arête générée, nous utilisons le nombre d'occurrences de chaque sommet dans l'hypergraphe initial. L'idée est qu'un sommet très présent a de fortes chances d'être dans une

Algorithme 1 HR (Hypergraph Reduction)

Entrée : un hypergraphe $\mathcal{H}$=(V, E) avec $|V|$=n et $|E|$=m

Sortie : l'hypergraphe réduit $\mathcal{H}_R$

1: // Etape 1 : calcul du nombre d'occurrences de chaque sommet dans les hyperarêtes de $\mathcal{H}$
2: **pour tout** $v \in V$ **faire**
3: $occur[v] \leftarrow 0$;
4: **fin pour**
5: **pour tout** $e \in E$ **faire**
6: **pour tout** $v \in e$ **faire**
7: $occur[v] \leftarrow occur[v] + 1$;
8: **fin pour**
9: **fin pour**
10: // Etape 2 : calcul des intersections des hyperarêtes et construction du graphe valué G=(V', E')
11: $V' \leftarrow \{v'_i$ avec $i = 1, \ldots, m\}$; // chaque $v'_i \in V'$ représente $e_i \in E$
12: $E' \leftarrow \{\}$;
13: **pour tout** $v'_i \cap v'_j \neq \emptyset$ **faire**
14: // intersection non vide entre e_i et e_j ($\in E$)
15: $E' \leftarrow E' \cup \{(v'_i, v'_j)\}$;
16: $w_{(v'_i,v'_j)} \leftarrow \sum_{v \in \{\psi^{-1}(v'_i) \cap \psi^{-1}(v'_j)\}} occur[v]$;
17: **fin pour**
18: // Etape 3 : sélection d'arêtes de G et génération de l'hypergraphe réduit $\mathcal{H}_R$=(V_R,E_R)
19: $V_R \leftarrow \{\}$;
20: $E_R \leftarrow \{\}$;
21: **tant que** $E' \neq \emptyset$ **faire**
22: Sélectionner $e'max = (v'max_i, v'max_j)$ ayant un poids w de valeur maximum
23: $V_R \leftarrow V_R \cup \{\psi^{-1}(v'max_i) \cap \psi^{-1}(v'max_j)\}$;
24: $E_R \leftarrow E_R \cup \{\{\psi^{-1}(v'max_i) \cap \psi^{-1}(v'max_j)\}\}$;
25: Suppression des arêtes $e' \in E'$ où $v'max_i$ ou $v'max_j$ est présent
26: **fin tant que**
27: **retourner** $\mathcal{H}_R$;

traverse minimales. Cela traduit un certain degré de transversalité. Si le nombre d'occurrences d'un sommet est égale au nombre d'hyperarêtes alors ce sommet forme à lui seul une traverse minimale. Notons que ceci est utilisé dans plusieurs algorithmes calculant des traverses [AvG09, BMR03, RS02]. Le poids d'une arête e'_k = (v'_i, v'_j), noté $w_{e'_k}$, est la somme des nombres d'occurrences des sommets présents dans l'intersection ayant entraîné la création de cette arête :

$$w_{e'_k} = \sum_{v \in \{\psi^{-1}(v'_i) \cap \psi^{-1}(v'_j)\}} occur[v]$$

Considérons l'exemple du tableau 1 comme un hypergraphe $\mathcal{H}$. Les transactions correspondent aux hyperarêtes e_k et les items aux sommets v_i. Dans un premier temps, le nombre d'occurrences de chaque sommet est évalué : $occur[A] = 3$, $occur[B] = 3$, $occur[C] = 5$, $occur[D] = 1$, $occur[E] = 4$, $occur[F] = 3$, $occur[G] = 2$ et $occur[H] = 4$. Ensuite, les intersections des hyperaretes sont calculées. Par exemple, $\psi^{-1}(v'_1) \cap \psi^{-1}(v'_3) = e_1 \cap e_3 = \{A, C, E\}$. La somme des nombres d'occurrences de A, C et E est égale à 12 ce qui correspond au poids de l'arête (v'_1, v'_3) produite dans G. Le matrice d'adjacence du graphe généré est :

$$\begin{pmatrix} 0 & 11 & 12 & 3 & 5 & 9 \\ 11 & 0 & 9 & 0 & 8 & 12 \\ 12 & 9 & 0 & 7 & 9 & 13 \\ 3 & 0 & 7 & 0 & 7 & 7 \\ 5 & 8 & 9 & 7 & 0 & 15 \\ 9 & 12 & 13 & 7 & 15 & 0 \end{pmatrix}$$

5.1.2 Etape 3 : génération de l'hypergraphe réduit

Après la construction du graphe valué, l'algorithme effectue une sélection d'arêtes par un algorithme glouton en sélectionnant l'arête ayant le plus fort poids tant qu'il reste des arêtes à selectionner. Chaque arête sélectionnée est transformée en une hyperarête pour l'hypergraphe réduit. Cette hyperarête contient les sommets de $\mathcal{H}$ correspondant à l'intersection des deux sommets de l'arête (qui sont des hyperarêtes dans $\mathcal{H}$). On obtient à la fin un ensemble d'hyperarêtes formant l'hypergraphe réduit $\mathcal{H}_R$=(V_R,E_R).

Reprenons notre exemple, l'arête (v'_5, v'_6) est sélectionnée car elle a le poids le plus élevé (i.e. 15). Les arêtes où figurent v'_5 ou v'_6 sont ensuite supprimées. Nous avons alors V_R = {B, C, F, H} et E_R = {{B, C, F, H}}). Le matrice d'adjacence du graphe restant est :

$$\begin{pmatrix} 0 & 11 & 12 & 3 & 0 & 0 \\ 11 & 0 & 9 & 0 & 0 & 0 \\ 12 & 9 & 0 & 7 & 0 & 0 \\ 3 & 0 & 7 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

La prochaine arête sélectionnée est (v'_1, v'_3). Après suppression des arêtes où figurent v'_1 ou v'_3, il n'y a plus d'arêtes donc l'algorithme termine.

L'hypergraphe réduit est donc $\mathcal{H}_R = (V_R, E_R)$ où V_R = {A, B, C, E, F, H} et E_R = {{A, C, E}, {B, C, F, H}}). Il y a 6 sommets au lieu de 8, et 2 hyperarêtes au lieu de 6.

5.2 Calcul des traverses minimales

La dernière étape consiste à calculer les traverses minimales de l'hypergraphe réduit. Ces traverses correspondent aux traverses minimales approchées de l'hypergraphe initial. Nous avons donc :

$$\widetilde{MinTr}(\mathcal{H}) = MinTr(\mathcal{H}_R)$$

Sur notre exemple, les traverses minimales de $\mathcal{H}_R$ sont $\{C,\ AB,\ AF,\ AH,\ BE,\ EF,\ EH\}$. Nous les considérons alors comme les traverses minimales approchées de $\mathcal{H}$. Les traverses minimales approchées de $\mathcal{H}$ sont donc $\{C, AB, AF, AH, BE, EF, EH\}$. Remarquons que les traverses minimales exactes de $\mathcal{H}$ sont : $\{AB, AC, CD, CF, CH, EF, EH, GH, AFG, BDE\}$.

6 Expérimentations

Les expérimentations ont été réalisées avec notre plateforme PODAMI [1] (Pattern Oriented DAta MIning) qui contient des implantations JAVA d'algorithmes de fouille de données orientée motifs.

6.1 Données et protocole

Quatre jeux de données ont été utilisés : Mushroom, Chess, Connect et Kosarak. Ils ont été récupérés sur le site de FIMI [2]. Mushroom contient des données sur des champignons et leur comestibilité. Chess contient des stratégies pour le jeu d'échecs. Connect contient des stratégies pour le jeu Puissance 4. Kosarak a été fourni à FIMI par Ferenc Bodon et contient des données anonymisées de navigation sur un portail de news hongrois.

Ces jeux de données (cf. table 2) ont été choisis pour couvrir les différents types de jeux de données existant selon deux classifications : [GZ01] et [FDMP10]. La classification proposée par [GZ01] (types 1, 2 , 3 et 4) se base sur la densité et la distribution de la bordure positive par rapport à la taille des motifs et par rapport à la valeur du seuil minimum de support. Remarquons qu'un jeu de données est dit "dense" lorsqu'il produit de longs motifs fréquents même pour des valeurs élevées de seuil minimum de support. La classification proposée par [FDMP10] (types I, II et III) étudie les deux bordures (positive et négative).

Nous avons suivi le protocole suivant. Pour chaque jeu de données et pour différentes valeurs de seuil minimum de support :

1) calcul de la bordure positive selon la valeur du support minimum, en utilisant IBE [SU03],
2) calcul de la bordure négative exacte (la référence) avec $Border\text{-}differential$ [DL05] (noté ici DL), de la bordure approchée avec δ-$MTminer$ (pour δ=1 et δ=2) et de la bordure approchée avec notre méthode notée HR,
3) dualisation vers les bordures positives (1 exacte et 3 approchées) en utilisant l'algorithme DL de calcul de traverses minimales.

Aux étapes 2 et 3, les statistiques suivantes sont calculées : nombre de motifs de la bordure calculée, taille moyenne d'un motif de la bordure calculée, et distance entre l'ensemble des motifs de la bordure calculée avec l'ensemble des motifs de la bordure exacte. Pour évaluer la distance entre deux bordures, nous nous sommes basés sur la distance de Karonski & Palka [KP77] qui utilise la distance de Hausdorff et qui a l'avantage d'être applicable à deux ensembles d'éléments de cardinalité différente. La distance "cosine" a été choisie pour déterminer la distance entre deux éléments (i.e. deux motifs). La distance D entre deux ensembles de motifs $\mathcal{X}$ et $\mathcal{Y}$ correspond à :

$$D(\mathcal{X},\mathcal{Y}) = \max\{\ h(\mathcal{X},\mathcal{Y})\ ,\ h(\mathcal{Y},\mathcal{X})\ \}$$
$$\text{avec}\ \ h(\mathcal{X},\mathcal{Y}) = \max_{X\in\mathcal{X}}\{\min_{Y\in\mathcal{Y}}\ d(X,Y)\}$$
$$\text{et}\ \ d(X,Y) = 1 - \frac{|X\cap Y|}{\sqrt{|X|\times|Y|}}$$

6.2 Résultats et discussion

Les figures 1, 2, 3 et 4 présentent, pour chaque jeu de données, le nombre de motifs et la taille moyenne des motifs des bordures négatives calculées en fonction du support minimum. Elles présentent aussi la distance entre les bordures négatives calculées et les bordures négatives exactes (produites par DL).

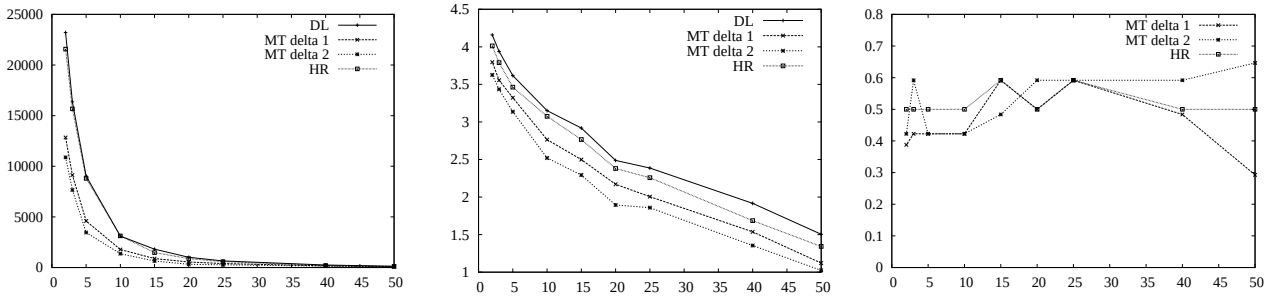
Nous pouvons observer que la cardinalité de $\widetilde{Bd^-(S)}$ est la plus faible pour 2-$MTminer$ sauf pour le jeu de données Connect où c'est alors HR. Pour Mushroom et Kosarak, le nombre de motifs de $Bd^-(S)$ produits par HR est très proche de celui de $Bd^-(S)$. Cependant, les motifs de $\widetilde{Bd^-(S)}$ sont différents d'après les distances observables.

Les motifs des $\widetilde{Bd^-(S)}$ sont plus courts que $Bd^-(S)$. Ils sont les plus courts pour 2-$MTminer$ sur chaque jeu de données. On observe même des motifs très courts pour δ-$MTminer$ sur Kosarak. Avec HR, les motifs produits sont légèrement plus courts que l'exact pour Mushroom et Kosarak. Pour Chess et Connect, HR et 1-$MTminer$ produisent des motifs de taille moyenne très proche.

1. *http://nicolas.durand.perso.luminy.univmed.fr*
2. Frequent Itemset Mining Implementations, *http://fimi.ua.ac.be/data/*

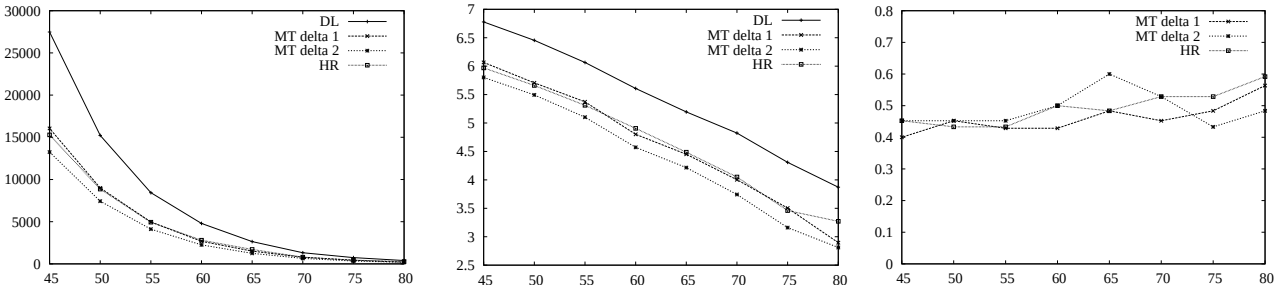
Jeu de données	Nbre transactions	Nbre items	Taille moyenne d'une transaction	Type selon Gouda & Zaki	Type selon Flouvat et al.
Mushroom	8124	119	23	4	II
Chess	3196	75	37	1	I
Connect	67557	129	43	2	II
Kosarak	990002	41270	8,1	3	III

TABLE 2 – Jeux de données utilisés



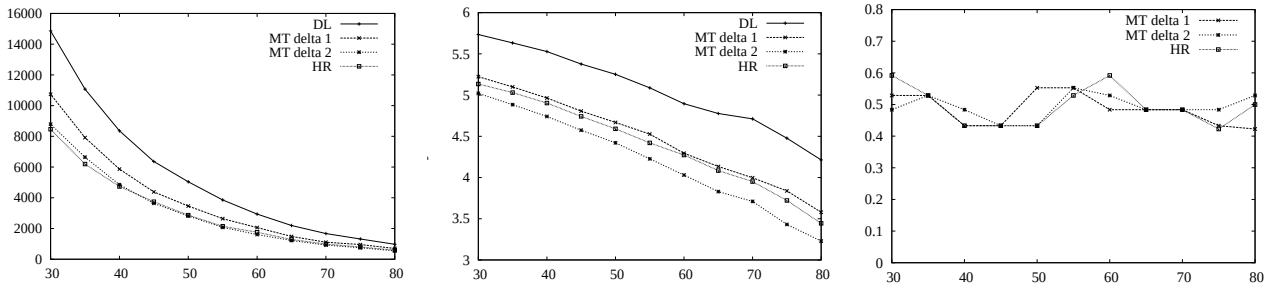
(a) Nombre de motifs (b) Taille moyenne des motifs (c) Distance avec $Bd^-(S)$

FIGURE 1 – Bordures négatives calculées sur Mushroom (en fonction du support minimum).



(a) Nombre de motifs (b) Taille moyenne des motifs (c) Distance avec $Bd^-(S)$

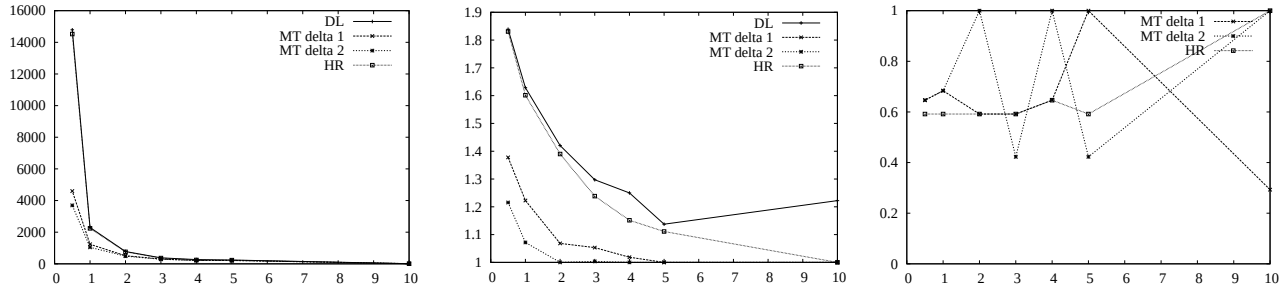
FIGURE 2 – Bordures négatives calculées sur Chess (en fonction du support minimum).



(a) Nombre de motifs (b) Taille moyenne des motifs (c) Distance avec $Bd^-(S)$

FIGURE 3 – Bordures négatives calculées sur Connect (en fonction du support minimum).

Concernant la distance entre $\widetilde{Bd^-(S)}$ et $Bd^-(S)$, 1-*MTminer* a produit les bordures approchées les plus proches pour Mushroom, Chess et Connect. Pour Kosarak, HR a obtenu les meilleurs résultats. 2-*MTminer* a obtenu de bons et de mauvais résultats. Toujours pour Kosarak, nous pouvons remarquer que les δ-*MTminer*



(a) Nombre de motifs (b) Taille moyenne des motifs (c) Distance avec $Bd^-(S)$

FIGURE 4 – Bordures négatives calculées sur Kosarak (en fonction du support minimum).

sont inconstants.

Les figures 5, 6, 7 et 8 présentent, pour chaque jeu de données, le nombre de motifs et la taille moyenne des motifs des bordures positives calculées en fonction du support minimum. Elles présentent aussi la distance entre les bordures postives calculées et les bordures positives exactes (produites par DL).

Le nombre de motifs de $\widetilde{Bd^+(S)}$ pour HR est le plus faible pour Mushroom, Chess et Connect. Ce n'est pas le cas pour Kosarak où 1-$MTminer$ et 2-$MTminer$ ont produit moins de motifs que HR. Cependant, ils ont échoué dans le sens où les distances avec les bordures positives exactes sont très élevées. Nous remarquons qu'avec HR, la taille de $\widetilde{Bd^+(S)}$ est bien inférieure à celle de $Bd^+(S)$.

Les motifs de $\widetilde{Bd^+(S)}$ générés par HR, sont plus longs que ceux de $Bd^+(S)$, et même plus longs que les δ-$MTminer$ (sauf pour Kosarak).

HR a permis d'obtenir les $\widetilde{Bd^+(S)}$ les plus proches de $Bd^+(S)$ pour Mushroom, Chess et Kosarak. Pour le jeu de données Connect, c'est moins évident car 1-$MTminer$ obtient aussi de bons résultats. Sur les quatre jeux de données, 1-$MTminer$ a produit de meilleurs résultats que 2-$MTminer$. Pour Kosarak, les deux δ-$MTminer$ ont échoué.

En conclusion de ces expérimentations, nous pouvons noté que notre méthode, HR, a permis de baisser la cardinalité des bordures positives produites, tout en gardant une distance raisonnable avec les bordures positives exactes. De plus, notre méthode semble robuste par rapport aux différents types de jeux de données que nous pouvons rencontrés.

7 Conclusion et perspectives

Nous avons proposé une nouvelle approche d'approximation des bordures de motifs fréquents via le calcul de traverses minimales d'hypergraphes. A partir de la bordure positive exacte, une bordure négative approchée est calculée et ensuite la bordure positive approchée correspondante est générée par dualisation. Le calcul proposé de traverses minimales approchées repose sur la réduction d'hypergraphes. Notre algorithme exploite pleinement les propriétés des hypergraphes formés lors de la dualisation vers la bodure négative. Nous avons montré de façon empirique que notre méthode produit une bordure positive approximative plus petite que la bordure positive exacte, tout en gardant une distance raisonnable avec elle.

Dans le futur, nous allons étudier l'utilisation de notre méthode pour introduire une part d'approximation dans des algorithmes qui se basent sur la dualisation [GKM+03, SU03, FDMP04]. Nous allons aussi développer des systèmes de recommandation utilisant les motifs de bordures positives approchées, dans le domaine du Web (services Web, navigation Web, ...).

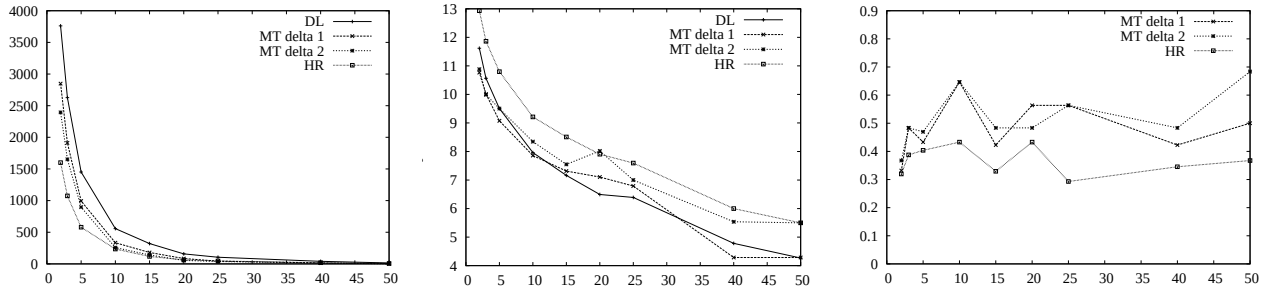
Références

[AGM04] F. Afrati, A. Gionis, and H. Mannila. Approximating a Collection of Frequent Sets. In *KDD'04*, pages 12–19, 2004.

[AIS93] R. Agrawal, T. Imielinski, and A. Swami. Mining Association Rules between Sets of Items in Large Database. *SIGMOD'93*, pages 207–216, 1993.

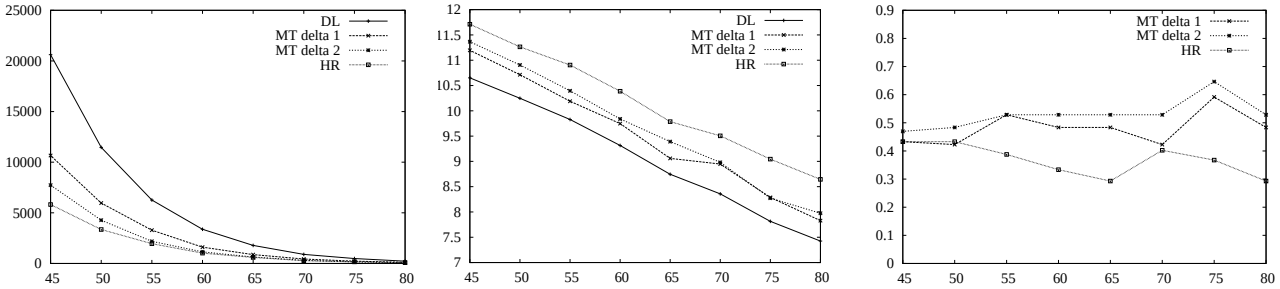
[AvG09] R. Abreu and A. van Gemund. A Low-Cost Approximate Minimal Hitting Set Algorithm and its Application to Model-Based Diagnosis. In *SARA'09*, 2009.

[BBR00] J. F. Boulicaut, A. Bykowski, and R. Rigotti. Approximation of Frequency Queries by Means of Free-sets. In *PKDD'00*, pages 75–85, 2000.



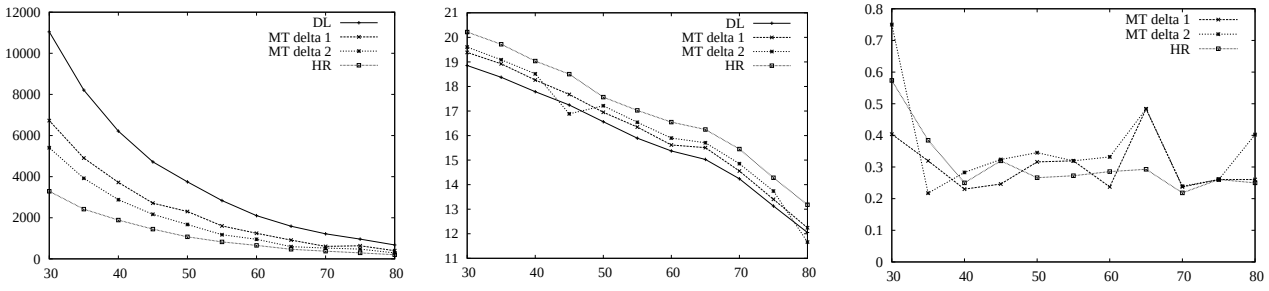
(a) Nombre de motifs (b) Taille moyenne des motifs (c) Distance avec $Bd^+(S)$

FIGURE 5 – Bordures positives calculées sur Mushroom (en fonction du support minimum).



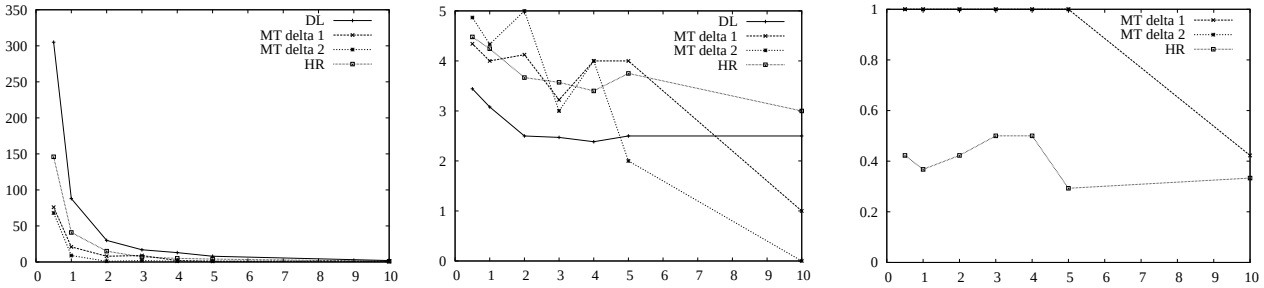
(a) Nombre de motifs (b) Taille moyenne des motifs (c) Distance avec $Bd^+(S)$

FIGURE 6 – Bordures positives calculées sur Chess (en fonction du support minimum).



(a) Nombre de motifs (b) Taille moyenne des motifs (c) Distance avec $Bd^+(S)$

FIGURE 7 – Bordures positives calculées sur Connect (en fonction du support minimum).



(a) Nombre de motifs (b) Taille moyenne des motifs (c) Distance avec $Bd^+(S)$

FIGURE 8 – Bordures positives calculées sur Kosarak (en fonction du support minimum).

[Ber89] C. Berge. *Hypergraphs : Combinatorics of Finite Sets*, volume 45. North Holland Mathematical Library, 1989.

[BMR03] J. Bailey, T. Manoukian, and K. Ramamohanarao. A Fast Algorithm for Computing Hypergraph Transversals and its Application in Mining Emerging Patterns. In *ICDM'03*, pages 485–488, 2003.

[Bol07] M. Boley. On Approximating Minimum Infrequent and Maximum Frequent Sets. In *DS'07*, pages 68–77, 2007.

[DBRL12] A. Ducournau, A. Bretto, S. Rital, and B. Laget. A Reductive Approach to Hypergraph Clustering : An Application to Image Segmentation. *Pattern Recognition*, 45(7) :2788–2803, 2012.

[DC02] N. Durand and B Crémilleux. ECCLAT : a New Approach of Clusters Discovery in Categorical Data. In *ES'02*, pages 177–190, Cambridge, UK, 2002.

[DL99] G. Dong and J. Li. Efficient Mining of Emerging Patterns : Discovering Trends and Differences. In *KDD'99*, pages 43–52, 1999.

[DL05] G. Dong and J. Li. Mining Border Descriptions of Emerging Patterns from DatasetPairs. *Knowledge and Information Systems*, 8(2) :178–202, 2005.

[DMP03] F. De Marchi and J.M. Petit. Zigzag : a New Algorithm for Mining Large Inclusion Dependencies in Database. In *ICDM'03*, pages 27–34, 2003.

[EG02] T. Eiter and G. Gottlob. Hypergraph Transversal Computation and Related Problems in Logic and AI. In *JELIA'02*, pages 549–564, 2002.

[FDMP04] F. Flouvat, F. De Marchi, and J.M. Petit. ABS : Adaptive Borders Search of Frequent Itemsets. In *FIMI'04*, 2004.

[FDMP10] F. Flouvat, F. De Marchi, and J.-M. Petit. A New Classification of Datasets for Frequent Itemsets. *Intelligent Information Systems*, 34 :1–19, 2010.

[FK96] M. L. Fredman and L. Khachiyan. On the Complexity of Dualization of Monotone Disjunctive Normal Forms. *Algorithms*, 21(3) :618–628, 1996.

[GKM+03] D. Gunopulos, R. Khardon, H. Mannila, S. Saluja, H. Toivonen, and R. S. Sharma. Discovering All Most Specific Sentences. *Database Systems*, 28(2) :140–174, 2003.

[GZ01] K. Gouda and M. J. Zaki. Efficiently Mining Maximal Frequent Itemsets. In *ICDM'01*, pages 163–170, 2001.

[HBC07] C. Hébert, A. Bretto, and B. Crémilleux. A data mining formalization to improve hypergraph transversal computation. *Fundamenta Informaticae*, 80(4) :415–433, 2007.

[KP77] M. Karonski and Z. Palka. One Standard Marczewski-Steinhaus Outdistances between Hypergraphs. *Zastosowania Matematyki Applicationes Mathematicae*, 16(1) :47–57, 1977.

[KS05] D. Kavvadias and E. Stavropoulos. An Efficient Algorithm for the Transversal Hypergraph Generation. *Graph Algorithms and Applications*, 9(2) :239–264, 2005.

[MT97] H. Mannila and H. Toivonen. Levelwise Search and Borders of Theories in Knowledge Discovery. *Data Mining and Knowledge Discovery*, 1(3) :241–258, 1997.

[PBTL99] N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Efficient Mining of Association Rules Using Closed Itemset Lattices. *Information Systems*, 24(1) :25–46, 1999.

[RBF05] K. Ramamohanarao, J. Bailey, and H. Fan. Efficient Mining of Contrast Patterns and Their Applications to Classification. In *ICISIP'05*, pages 39–47, 2005.

[RS02] D. P. Ruchkys and S. W. Song. A Parallel Approximation Hitting Set Algorithm for Gene Expression Analysis. In *SBACPAD'02*, pages 75–81, 2002.

[RZC10] F. Rioult, B. Zanuttini, and B. Crémilleux. Nonredundant Generalized Rules and Their Impact in Classification. *Advances in Intelligent Information Systems*, 265 :3–25, 2010.

[SU03] K. Satoh and T. Uno. Enumerating Maximal Frequent Sets Using Irredundant Dualization. In *DS'03*, pages 256–268, 2003.

[VØ00] S. Vinterbo and A. Øhrn. Minimal Approximate Hitting Sets and Rule Templates. *Approximate Reasoning*, 25 :123–143, 2000.