Symétries et Extraction de Motifs Ensemblistes

Said Jabbour*, Mehdi Khiari**, Lakhdar Sais*, Yakoub Salhi *, Karim Tabia*

*CRIL UMR CNRS 8188, Université d'Artois, F-62300 Lens,
{jabbour, sais, salhi, tabia}@cril.univ-artois.fr,
http://www.cril.univ-artois.fr
**Zero To One Technology, Campus Effiscience, F-14460 Colombelles,
mehdi.khiari@zto-technology.com,
http://zto-technology.com

Résumé. Les symétries sont des propriétés structurelles qu'on détecte dans un grand nombre de bases de données. Dans cet article, nous étudions l'exploitation des symétries pour élaguer l'espace de recherche dans les problèmes d'extraction de motifs ensemblistes. Notre approche est basée sur une intégration dynamique des symétries dans les algorithmes de type Apriori permettant de réduire l'espace des motifs candidats. En effet, pour un motif donné, les symétries nous permettent de déduire les motifs qui lui sont symétriques et vérifiant par conséquent les mêmes propriétés. Nous détaillons notre approche en utilisant l'exemple des motifs fréquents. Ensuite, nous la généralisons au cadre unificateur de Mannila et Toivonen pour l'extraction des motifs ensemblistes. Les expériences menées montrent la faisabilité et l'apport de notre approche d'élagage basé sur les symétries.

1 Introduction

Le but de cet article est d'introduire un cadre d'élagage basé sur les symétries pour la fouille de données. Dans d'autres domaines, tels que la programmation par contraintes et la satisfaisabilité propositionnelle, les symétries sont souvent exploitées pour élaguer l'espace de recherche et améliorer les performances des solveurs. Notre but est de montrer l'apport d'un élagage à base de symétries dans un cadre de fouille de données. Dans cet article, nous nous intéressons au cas de l'extraction de motifs ensemblistes et nous prenons comme exemple d'application l'algorithme APRIORI (Agrawal et Srikant, 1994).

Les symétries sont un concept fondamental en informatique, mathématiques, physiques et plein d'autres domaines. Elles existent dans divers problèmes réels. Les symétries sont communément exploitées dans la résolution de problèmes combinatoires, tels que les problèmes d'ordonnancement. Par exemple, dans un problème d'ordonnancement où certaines machines peuvent être interchangeables, partant d'un ordonnancement valide, on peut en obtenir un autre valide en permutant ces machines. Exploiter les symétries permet de réduire le coût de la recherche de solution, en évitant d'explorer des branches symétriques de l'espace de recherche. Beaucoup de travaux ont ainsi porté sur les symétries dans les problèmes de satisfaction de contraintes (CSP) (e.g. (Puget, 1993; Gent et Smith, 2000)), satisfaisabilité propositionnelle

(e.g. (Benhamou et Saïs, 1994; Crawford et al., 1996)) et recherche opérationnelle (e.g. (Margot, 2003; Liberti, 2012)).

En fouille de données, les symétries sont principalement étudiées en fouille de graphes (e.g. (Desrosiers et al., 2007; Vanetik, 2010)). Récemment, (Jabbour et al., 2012) a proposé un nouveau cadre pour casser les symétries dans les problèmes de fouille de données. Dans (Jabbour et al., 2012), les symétries représentent des permutations entre les items laissant invariant le jeu de données. L'exploitation des symétries dans ce cadre s'effectue en pré-traitement permettant de supprimer des items du jeu de données initial. Cette approche est différente de celle utilisée dans les problèmes CSP et SAT où les symétries sont éliminées en ajoutant au réseau de contraintes, des contraintes supplémentaires appelées prédicats d'élimination de symétries.

(Jabbour et al., 2012), mentionne aussi que les symétries peuvent être intégrées dans les algorithmes de type APRIORI pour l'élagage de l'espace de recherche. Dans cet article, nous explorons en profondeur cette piste. Notre motivation vient du fait que l'élagage basé sur les symétries peut être généralisé à d'autres tâches de fouille de données ce qui n'est pas le cas pour l'approche présentée dans (Jabbour et al., 2012). Notre approche nous permet aussi d'exploiter efficacement les symétries présentes au sein d'une même transaction, ce qui n'était pas le cas avec l'approche de (Jabbour et al., 2012).

Nous avons choisi l'algorithme APRIORI, algorithme pionnier de l'extraction de motifs fréquents et de règles d'association, comme exemple pour montrer la faisabilité de notre approche d'élagage à base de symétries. Les expérimentations menées montrent la faisabilité et l'efficacité de notre approche. À la limite de nos connaissances, notre approche est la première approche opérationnelle permettant l'exploitation des symétries dans les problèmes d'extraction de motifs ensemblistes.

2 Définitions

2.1 Extraction de motifs

Soit $\mathcal{I}$ un ensemble d'*items* et $\mathcal{T}$ un ensemble d'*identifiants de transactions*. Un ensemble $I \subseteq \mathcal{I}$ est appelé *itemset* ou *motif*. Une *transaction* est un couple (tid, I) où tid est un identifiant de transaction ($tid \in \mathcal{T}$) et I est un itemset ($I \in \mathcal{I}$). Une *base de données transactionnelle* $\mathcal{D}$ est un ensemble fini de transactions ayant chacune un identifiant unique. Nous notons $\mathcal{T}_{id}(\mathcal{D}) = \{tid | (tid, I) \in \mathcal{D}\}$ l'ensemble des identifiants des transactions de $\mathcal{D}$. $\mathcal{I}_{items}(O)$ dénote l'ensemble de tous les items appartenant à l'objet syntaxique O (e.g. base de données transactionnelle, motif, etc). On dit que la transaction (tid, I) *supporte* un motif J si $J \subseteq I$.

La *couverture* d'un motif I dans $\mathcal{D}$ est l'ensemble des transactions de $\mathcal{D}$ supportant I : $\mathcal{C}over(I, \mathcal{D}) = \{tid \mid (tid, J) \in \mathcal{D} \text{ et } I \subseteq J\}$. Le *support* d'un motif I dans $\mathcal{D}$ est défini par : $\mathcal{S}upp(I, \mathcal{D}) = |\mathcal{C}over(I, \mathcal{D})|$. Enfin, la fréquence de I dans $\mathcal{D}$ est définie par : $\mathcal{F}req(I, \mathcal{D}) = \frac{\mathcal{S}upp(I,\mathcal{D})}{|\mathcal{D}|}$.

Exemple 1 *Soit le jeu de données transactionnel $\mathcal{D}$ (cf. Table 1) où $\mathcal{I} = \{A, B, C, D, E, F\}$. On a $\mathcal{S}upp(\{B, D\}, \mathcal{D})$= $|\{002, 007\}|$=2 et $\mathcal{F}req(\{B, D\}, \mathcal{D})$=$\frac{2}{9}$.*

Dans la suite, nous notons par λ un seuil de support minimal, avec $0 \leq \lambda \leq |\mathcal{D}|$. L'ensemble des motifs fréquents dans $\mathcal{D}$ est défini comme suit :

$$\mathcal{FIM}(\mathcal{D}, \lambda) = \{I \subseteq \mathcal{I} \mid \mathcal{S}upp(I, \mathcal{D}) \geq \lambda\}$$

tid	itemset					
001	A	B			E	F
002	A	B	C	D		
003			C	D	E	F
004	A		C			
005	A				E	
006			C		E	
007		B		D		
008		B				F
009				D		F

TAB. 1: Exemple de jeu de données transactionnelles

2.2 Symétries et extraction de motifs fréquents

Intuitivement, une symétrie dans $\mathcal{D}$ est une permutation d'items laissant $\mathcal{D}$ invariant. Une telle permutation d'items induit une permutation sur les transactions. Le jeu de données demeurant tout de même inchangé, les propriétés d'un motif sont alors préservées par la symétrie. L'ensemble des symétries forme un groupe de symétries, dont l'élément neutre est toute permutation associant un item à lui même. Dans ce qui suit, nous utilisons des notations de la théorie des groupes.

Commençons par définir formellement les symétries dans le cadre de l'extraction de motifs fréquents.

Définition 1 (Permutation) *Une permutation p d'un ensemble fini $\mathcal{S}$ est une bijection de $\mathcal{S}$ vers $\mathcal{S}$.*

Chaque permutation p peut être représentée par un ensemble de cycles, $c_1 \dots c_n$ où chaque cycle $c_i = (a_1, \dots, a_k)$ est une liste d'éléments de $\mathcal{S}$ tel que $p(a_j) = a_{j+1}$ pour $j = 1, \dots, k-1$, et $p(a_k) = a_1$. Dans la suite, pour des raisons de clarté, nous omettons les cycles de la forme (a, a).

Nous notons par $\mathcal{P}(\mathcal{I}_{items}(\mathcal{D}))$ (resp. $\mathcal{P}(\mathcal{T}_{id}(\mathcal{D}))$) l'ensemble de toutes les permutations dans $\mathcal{I}_{items}(\mathcal{D})$ (resp. dans $\mathcal{T}_{id}(\mathcal{D})$).

Nous notons par $\mathcal{P}(\mathcal{T}_{id}(\mathcal{D}), \mathcal{I}_{items}(\mathcal{D}))$ l'ensemble des permutations : $\{\sigma \circ f \mid \sigma \in \mathcal{I}_{items}(\mathcal{D})\ et\ f \in \mathcal{T}_{id}(\mathcal{D})\}$. $(\mathcal{P}(\mathcal{T}_{id}(\mathcal{D}), \mathcal{I}_{items}(\mathcal{D})), \circ)$ est un groupe dont l'élément neutre est la permutation identité.

Une structure $(S', \circ)$ est un sous groupe du groupe $(\mathcal{P}(\mathcal{T}_{id}(\mathcal{D}), \mathcal{I}_{items}(\mathcal{D})), \circ)$ si et seulement si $(S', \circ)$ est un groupe et $S' \subseteq \mathcal{P}(\mathcal{T}_{id}(\mathcal{D}), \mathcal{I}_{items}(\mathcal{D}))$.

Une permutation $\sigma \circ f$ est appliquée à une jeu de données $\mathcal{D}$ comme suit : $\sigma \circ f(\mathcal{D}) = \{(f(tid), \sigma(I)) | (tid, I) \in \mathcal{D}\}$ où $\sigma(I) = \{\sigma(a) | a \in I\}$.

Définition 2 (Symétrie) *Une symétrie de $\mathcal{D}$ est une permutation $\sigma \circ f$ dans $\mathcal{P}(\mathcal{T}_{id}(\mathcal{D}), \mathcal{I}_{items}(\mathcal{D}))$ tel que $\sigma \in \mathcal{P}(\mathcal{I}_{items}(\mathcal{D}))$, $f \in \mathcal{P}(\mathcal{T}_{id}(\mathcal{D}))$ et $\sigma \circ f(\mathcal{D}) = \mathcal{D}$.*

Exemple 2 *Considérons le jeu de données $\mathcal{D}$ de la Table 1. Soit $\sigma = (A, C)(B, D)$ une permutation dans $\mathcal{I}_{items}(\mathcal{D})$ et $f = (001, 003)(005, 006)(008, 009)$ une permutation dans $\mathcal{T}_{id}(\mathcal{D})$. La permutation $\sigma \circ f$ est une symétrie de $\mathcal{D}$.*

Soit $\mathcal{S}(\mathcal{D})$ l'ensemble de toutes les symétries de $\mathcal{D}$. Il est à noter que $(\mathcal{S}(\mathcal{D}), \circ)$ est un sous groupe de $(\mathcal{P}(\mathcal{T}_{id}(\mathcal{D}), \mathcal{I}_{items}(\mathcal{D})), \circ)$.

Comme l'ensemble des transactions est invariant par symétrie, toutes les propriétés d'un motif telles que la fréquence, la fermeture et la maximalité sont préservées. La proposition suivante montre que la symétrie préserve la fréquence d'un motif.

Proposition 1 *Soit $\mathcal{D}$ un jeu de données transactionnelles, $\sigma \circ f$ une symétrie dans $\mathcal{D}$, λ un seuil de support minimal et I un motif. $I \in \mathcal{FIM}(\mathcal{D}, \lambda)$ ssi $\sigma(I) \in \mathcal{FIM}(\mathcal{D}, \lambda)$.*

Preuve 1 *$\sigma \circ f$ étant une symétrie, le support de I est égal à celui de $\sigma(I)$. Par conséquent, $I \in \mathcal{FIM}(f(\mathcal{D}), \lambda)$ ssi $\sigma(I) \in \mathcal{FIM}(\mathcal{D}, \lambda)$.*

2.3 Détection de symétries dans les bases de données transactionnelles

Un moyen commun pour détecter les symétries est d'encoder les données sous forme de graphe et de rechercher les automorphismes de graphe. La plupart des outils de détection de symétries se basent principalement sur les automorphismes d'un graphe coloré. Les couleurs des sommets sont utilisées pour contraindre le lien avec des sommets de la même couleur, i.e. deux sommets de la même couleur peuvent être permutés. La première implémentation pour calculer les automorphismes de graphes, appelée `nauty`, est proposée dans (McKay, 1981). D'autres implémentations améliorées utilisant des heuristiques d'élagage provenant du domaine de la théorie des groupes sont présentées dans (Aloul et al., 2003; Junttila et Kaski, 2007).

Ainsi, pour détecter les symétries dans un jeu de données transactionnel $\mathcal{D}$, il suffit d'encoder $\mathcal{D}$ sous forme de graphe coloré $\mathcal{G}$ de telle sorte que les symétries dans $\mathcal{D}$ correspondent aux automorphismes de $\mathcal{G}$.

Définition 3 *Un graphe coloré est un triplet $\mathcal{G} = (V, E, \eta)$ où V est l'ensemble de ses sommets, $E \in V \times V$ l'ensemble de ses arêtes et η est une fonction de V dans N qui associe un entier positif (une couleur) à chaque sommet.*

Définition 4 *Une permutation σ de V est un automorphisme de $\mathcal{G}$ ssi $\sigma(\mathcal{G}) = \mathcal{G}$ ou d'une manière équivalente $\sigma(E) = E$.*

Nous montrons maintenant comment un jeu de données transactionnelles peut être encodé sous forme de graphe coloré non orienté.

Définition 5 (De $\mathcal{D}$ à $\mathcal{G}$) *Nous définissons le graphe coloré non orienté $\mathcal{G}$ associé à $\mathcal{D}$ par $\mathcal{G}(\mathcal{D}) = (V, E, \eta)$ où $V = \mathcal{I} \cup \mathcal{T}_{id}(\mathcal{D})$, $E = \{(tid, i) | \exists (tid, I) \in \mathcal{D}, i \in I\}$ et $\forall v \in V$,*

$$\eta(v) = \begin{cases} 0 & \text{si } v \in \mathcal{I}_{items}(\mathcal{D}) \\ 1 & \text{si } v \in \mathcal{T}_{id}(\mathcal{D}) \end{cases}$$

La figure 1 décrit la conversion de $\mathcal{D}$ (cf. Table 1) en un graphe coloré $\mathcal{G}(\mathcal{D})$. Dans cette figure, les items sont représentés par des cercles (couleur 1), alors que les identifiants des transactions sont représentés par de rectangles (couleur 2).

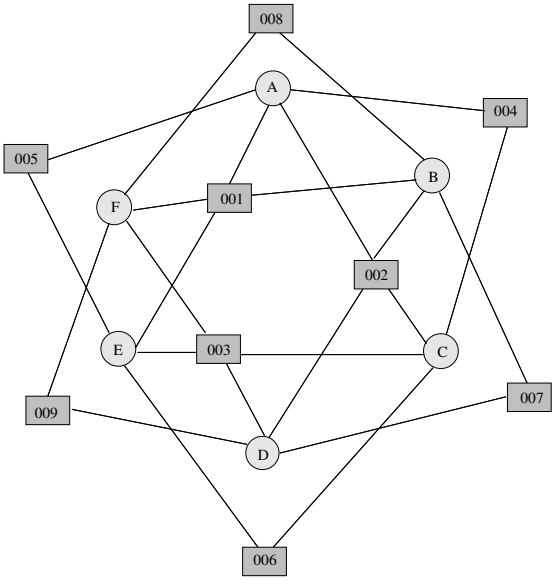

FIG. 1: Graphe $\mathcal{G}(\mathcal{D})$

Grâce à cette transformation, les symétries dans $\mathcal{D}$ correspondent aux automorphismes dans $\mathcal{G}(\mathcal{D})$.

Étant donné un graphe $\mathcal{G}(\mathcal{D})$ et une partition initiale π, on peut trouver, en utilisant NAUTY (McKay, 1981), les automorphismes laissant $\mathcal{G}(\mathcal{D})$ invariant. Par exemple, $A = (A, C)(B, D)$ $(001, 003)(005, 006)(008, 009)$ est un automorphisme de $\mathcal{G}(\mathcal{D})$. La symétrie correspondante est $\sigma \circ f$ où $\sigma = (A, C)(B, D)$ et $f = (001, 003)(005, 006)(008, 009)$.

3 Élagage à base de symétries

Dans cette section, nous montrons comment les symétries peuvent être exploitées afin d'élaguer l'espace de recherche dans le problème d'extraction de motifs fréquents. Nous utilisons l'exemple de l'algorithme APRIORI (Agrawal et Srikant, 1994) (cf. Algorithme 1). Cet algorithme se base sur la propriété d'anti-monotonie : si un motif est fréquent alors toutes ses généralisations le sont aussi. Il procède par une recherche par niveau des motifs fréquents. En effet, il commence par extraire l'ensemble F_1 des motifs fréquents de taille 1 (ligne 1). Puis il calcule itérativement les ensembles de F_2 (fréquents de taille 2) jusqu'à (F_n) tel que $F_{n+1} = \emptyset$ (lignes 2-8).

Nous décrivons dans ce qui suit comment nous avons intégré notre approche d'élagage à base de symétries dans l'algorithme APRIORI. Notre approche contribue essentiellement à réduire l'ensemble des motifs candidats tout comme la propriété d'anti-monotonie. Notre approche est décrite dans l'algorithme 2.

Algorithme 1 : APRIORI

Données : D : jeu de données, λ : seuil de support minimal
Sortie : l'ensemble de tous les motifs fréquents dans D

1 $F_1 \leftarrow \{$motifs fréquents de taille 1$\}$;
2 **pour** $(k = 2\,;\, F_{k-1} \neq \emptyset\,;\, k++)$ **faire**
3 $\quad F_k \leftarrow \emptyset$;
4 $\quad C_k \leftarrow CandidatesGen(F_{k-1})$;
5 $\quad$ **pour** $(c \in C_k)$ **faire**
6 $\quad\quad supp(c) \leftarrow CalculSupp(c, \mathcal{D})$;
7 $\quad\quad$ **si** $(supp(c) \geq \lambda)$ **alors**
8 $\quad\quad\quad F_k \leftarrow F_k \cup \{c\}$;
9 **retourner** $(\bigcup_k F_k)$;

Algorithme 2 : APRIORI$_{Sym}$

Données : D : jeu de données, λ : seuil de support minimal, $\mathcal{S}$: symétries dans D
Sortie : l'ensemble de tous les motifs fréquents dans D

1 $F_1 \leftarrow \{$motifs fréquents de taille 1$\}$;
2 **pour** $(k = 2\,;\, F_{k-1} \neq \emptyset\,;\, k++)$ **faire**
3 $\quad F_k \leftarrow \emptyset$;
4 $\quad C_k \leftarrow CandidatesGen(F_{k-1})$;
5 $\quad$ **pour** $(c \in C_k)$ **faire**
6 $\quad\quad supp(c) \leftarrow CalculSupp(c, \mathcal{D})$;
7 $\quad\quad S_c \leftarrow SymmGen(c, \mathcal{S})$;
8 $\quad\quad$ **si** $(supp(c) \geq \lambda)$ **alors**
9 $\quad\quad\quad F_k \leftarrow F_k \cup \{c\} \cup S_c$;
10 $\quad\quad C_k = C_k \setminus S_c$;
11 **retourner** $(\bigcup_k F_k)$;

Description de l'algorithme : Similairement à APRIORI, l'entrée de APRIORI$_{Sym}$ est composée de : un jeu de données transactionnelles $\mathcal{D}$, un seuil de support minimal λ, et un argument supplémentaire : l'ensemble $\mathcal{S}$ des symétries dans $\mathcal{D}$. Il retourne l'ensemble de tous les motifs fréquents dans D.

Lignes 1-4. Cette partie est similaire à APRIORI. Néanmoins, l'ensemble des symétries $\mathcal{S}$ est utilisé pour améliorer le calcul des itemsets fréquents (ligne 1). Par exemple, si $\{a\}$ est fréquent (resp. non fréquent) alors, pour tout $\sigma \circ f \in \mathcal{S}$, le motif $\{\sigma(a)\}$ est aussi un motif fréquent (resp. non fréquent).

Lignes 5-10. Dans cette boucle, nous commençons par calculer le support d'un candidat c (ligne 6) et l'ensemble S_c des candidats symétriques à c (ligne 7). Si c est fréquent (ligne 8), alors tous les éléments de S_c le sont aussi (ligne 9). Ensuite, les éléments de S_c sont retirés de l'ensemble des candidats (ligne 10).

Proposition 2 *L'algorithme* APRIORI$_{Sym}$ *est correct.*

Preuve 2 *C'est une conséquence immédiate du fait qu'*APRIORI *est un algorithme correct et de la Proposition 1.*

Par exemple, soit $\mathcal{D}$ une base de données transactionnelles telle que $\mathcal{I}_{items}(\mathcal{D}) = \{A, B, C, D\}$ et $\sigma \circ f$ est une symétrie telle que $\sigma = (A, D)(B, C)$. Nous supposons que les motifs $\{A\}$, $\{B\}$, $\{C\}$ et $\{D\}$ sont fréquents. Nous supposons aussi que, dans la ligne 8, nous obtenons que $\{A, B\}$ est non fréquent. En utilisant la propriété d'anti-monotonie, nous obtenons que les motifs $\{A, B, C\}$, $\{A, B, D\}$ et $\{A, B, C, D\}$ sont non fréquents. De plus, en utilisant σ, nous obtenons $\{C, D\} = \sigma(\{A, B\})$. Ainsi, par anti-monotonie, $\{A, C, D\}$ et $\{B, C, D\}$ son aussi non fréquents. Cet exemple est illustré par la Figure 2 où les nœuds décrits par des cercles représentent les motifs élagués grâce à notre approche d'élagage à base de symétries.

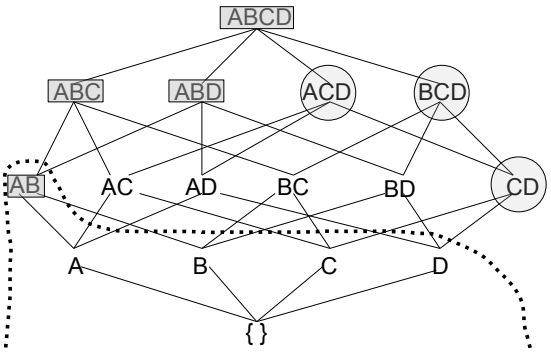

FIG. 2: Élagage à base de symétries

4 Étude expérimentale

4.1 Contexte expérimental

Nous présentons dans cette section quelques résultats expérimentaux montrant la faisabilité et l'apport de notre approche.

Nous présentons dans ce qui suit des expérimentations menées sur des jeux de données du répertoire de l'UCI [1] et le jeu de données réel BMS-WebView-2 (Zheng et al., 2001). Ce je de données contient des données sur les flux de clics enregistrés sur des sites de e-commerce. La Table 2 fournit pour chaque jeu de données son nombre de transactions (#trans), nombre d'items distincts (#items) et sa densité : la densité étant définie par le rapport entre la taille moyenne des transactions et #items.

Pour détecter les symétries dans les jeux de données, nous utilisons Saucy [2]. Dans la plupart des cas, Saucy permet de détecter les symetries dans les jeux de données en moins d'une seconde. La Table 3 récapitule les temps d'extraction des symétries dans les jeux de données utilisés dans notre expérimentation.

1. `http://www.ics.uci.edu/~mlearn/MLRepository.html`
2. Saucy2 : Fast symmetry discovery, `http://vlsicad.eecs.umich.edu/BK/SAUCY/`

bdd	#trans	#items	densité
Zoo	101	43	39%
Mushroom	8 142	117	18%
Australian	690	55	25%
Solar flare	323	40	32%
BMS-WebView-2	77 512	3 341	0.14%
Letter-rec	20 000	74	23%

TAB. 2: Description des jeux de données utilisés.

bdd	#sym	temps (s)
Zoo	2	0.02
Mushroom	10	0.94
Australian	2	0.06
Solar flare	2	0.05
BMS-WebView-2	10	4.31
Letter-rec	0	2.12

TAB. 3: Temps d'extraction des symétries.

Les symétries sont présentes dans les jeux de données indépendamment de toute tâche d'extraction de motifs, ainsi notre approche d'élagage basée sur les symétries procède en deux étapes. Nous commençons par extraire les symétries en utilisant Saucy. Ensuite, l'entrée de notre algorithme APRIORI$_{Sym}$ sera composée de l'entrée standard de l'algorithme APRIORI (i.e. jeu de données et un seuil de fréquence minimale) à laquelle on ajoute l'ensemble des symétries extraites.

Afin de quantifier le gain obtenu par APRIORI$_{Sym}$, nous introduisons les deux mesures suivantes :

- $\mathcal{M}_{TA} = \frac{temps_{exec}(\text{APRIORI}) - temps_{exec}(\text{APRIORI}_{Sym})}{temps_{exec}(\text{APRIORI})}$: permet de quantifier le gain obtenu en terme de temps d'exécution. Notons que cette mesure ne prend pas en considération le temps nécessaire pour extraire les symétries étant donné que cette opération n'est effectuée qu'une et une seule fois pour chaque jeu de données considéré.
- $\mathcal{M}_{SR} = \frac{\#\text{bdd scans}(\text{APRIORI}) - \#\text{bdd scans}(\text{APRIORI}_{Sym})}{\#\text{bdd scans}(\text{APRIORI})}$: permet de quantifier le gain obtenu en terme de réduction du nombre d'opérations de calcul de fréquences et, par conséquent, du nombre de scans de la base de données.

4.2 Résultats expérimentaux

Nous avons choisi de montrer l'apport de l'exploitation des symétries pour l'élagage de l'espace de recherche via l'algorithme historique d'extraction de motifs fréquents APRIORI (cf. Algorithme 1). Nous nous focalisons principalement sur l'apport des symétries en terme de réduction du nombre d'accès à la base de données ainsi qu'en terme d'accélération du processus d'extraction.

Notons que dans les jeux de données réels, nous détectons peu de symétries (cf. Table 3). En effet, dans ces jeux, on a tendance à détecter beaucoup plus de symétries locales (symétries dans une portion de $\mathcal{D}$) que globales (symétries dans tout $\mathcal{D}$). Les expérimentations menées dans cette section montrent que même avec très peu de symétries détectées, nous arrivons à obtenir un élagage significatif.

La Table 4 montre l'efficacité de l'exploitation des symétries pour améliorer l'élagage dans l'algorithme APRIORI même avec très peu de symétries détectées. Nous notons par #freq$_{Sym}$ (resp. #infreq$_{Sym}$) le nombre de motifs fréquents (resp. non fréquents) déduits par symétries. Par exemple, pour le jeu de données Australian ne contenant que 2 symétries, et un seuil de fréquence minimale $minfr = 1\%$, nous observons ce qui suit : en utilisant APRIORI, 480.000 opérations de calcul de fréquence sont effectuées. Alors qu'en utilisant APRIORI$_{Sym}$ les propriétés (fréquent, non fréquent) d'environ 115.000 motifs (#freq$_{Sym}$+#infreq$_{Sym}$) sont déduites par symétries (cf. Table 4). Dans ce même cas, APRIORI$_{Sym}$ est 23% plus rapide que APRIORI.

Jeu de données	$minfr$	Sym Pruning	#db scans	$\mathcal{M}_{SR}$	#freq	#freq$_{Sym}$	#infreq$_{Sym}$	$\mathcal{M}_{\mathcal{TA}}$
Australian	1%	−−	479 402	24%	426 763	−−	−−	23%
		✓	364 822			100 961	13 613	
	5%	−−	28 535	22%	20 386	−−	−−	22%
		✓	22 288			4.461	1 886	
Solar-Flare	1%	−−	147 270	9%	145 893	−−	−−	8%
		✓	133 056			14 128	1 291	
	15%	−−	5 684	8%	5 495	−−	−−	9%
		✓	5 203			448	33	
Mushroom	5%	−−	3 764 532	0.4%	3 755 511	−−	−−	6%
		✓	3 748 084			16 384	8 957	
Zoo	5%	−−	587 782	20%	486 099	−−	−−	15%
		✓	487 224			100 480	78	
	15%	−−	103 318	23%	102 440	−−	−−	17%
		✓	79 492			23 776	50	
BMS-WebView-2	1%	−−	4 908	0%	81	−−	−−	0%
		✓	4 908			0	0	
Letter-recognition	5%	−−	32 680	0%	15 719	−−	−−	0%
		✓	32 680			0	0	

TAB. 4: Résultats expérimentaux

Un autre résultat important, est que l'algorithme APRIORI$_{sym}$ garde les mêmes performances qu'APRIORI lorsqu'il s'agit de jeux de données ne contenant pas de symétries (cf. Table 4). Ainsi, notre approche d'élagage basée sur les symétries peut être généralisée à d'autres algorithmes d'extraction de motifs leur permettant de tirer profit d'une éventuelle présence de symétries dans les jeux de données. Nous montrons dans la section 5 la faisabilité de cette généralisation.

Notons que le jeu de données BMS-WebView-2 contient 10 symétries. Nous n'observons pourtant aucune influence de ces symétries sur l'élagage de l'espace de recherche. En effet, l'efficacité de l'élagage par symétrie n'est pas proportionnelle au nombre de symétries détectées : si les symétries détectées n'apparaissent que dans très peu de transactions, comme c'est le cas pour BMS-WebView-2, tous les motifs les impliquant sent non fréquents et sont alors efficacement élagués par APRIORI.

4.3 Discussion

Les résultats expérimentaux montrent que notre approche d'élagage à base de symétries intégrée à l'algorithme APRIORI permet d'améliorer considérablement les performances de ce dernier. Pour un jeu de données ne contenant pas de symétries, APRIORI$_{sym}$ a les même performances que APRIORI. Ceci est expliqué par le fait que les symétries sont extraites une et une seule fois pour un jeu de données donné. Le gain en terme de temps d'exécution et ainsi toujours supérieur ou égal à zéro ($\mathcal{M}_{TA}$(APRIORI $_{Sym}$,APRIORI) ≥ 0).

Les symétries sont ainsi une propriété importante dans les données qui devrait être considérée dans le but d'améliorer les performances des algorithmes d'extraction de motifs actuels sur certaines familles de jeux de données.

5 Généralisation de l'utilisation des symétries

Dans cette section, nous proposons une généralisation de notre approche à d'autres problèmes d'extraction de motifs. Nous nous plaçons dans le cadre unificateur de (Mannila et Toivonen, 1997). Soit $\mathcal{D}$ une base de données transactionnelle, $\mathcal{L}$ un langage décrivant les données (un ensemble de motifs), et q un prédicat d'intérêt permettant d'évaluer si un élément de $\mathcal{L}$ est une information intéressante. Dans ce contexte, nous considérons qu'une tâche d'extraction de motifs consiste à calculer l'ensemble suivant : $\mathcal{TH}(\mathcal{D}, \mathcal{L}, q) = \{a \in \mathcal{L} \mid q(\mathcal{D}, a)\ est\ vraie\}$.

Par exemple, le problème d'énumération de motifs fréquents peut être décrit en définissant $\mathcal{L}$ comme étant l'ensemble de tous les sous ensembles non vides de l'ensemble d'items de $\mathcal{D}$ et q le prédicat imposant que chaque sous ensemble ait un support supérieur à un seuil donné.

Un prédicat d'intérêt q est dit stable par symétrie, si étant donnée une base de données transactionnelle $\mathcal{D}$, pour toute symétrie σ, $q(\mathcal{D}, a)$ est vérifié ssi $q(\mathcal{D}, \sigma(a))$ l'est aussi.

Ainsi, pour tout prédicat d'intérêt stable par symétrie, les symétries peuvent être exploitées pour élaguer l'espace de recherche. En effet, si un élément a du langage est jugé intéressant (resp. non intéressant), alors $\sigma(a)$ est aussi intéressant (resp. non intéressant), pour toute symétrie σ. Ainsi, notre approche peut être intégrée dans différents problèmes d'extraction de motifs où les prédicats d'intérêts sont stables par symétrie.

Pour illustrer cette généralisation, considérons le problème d'énumération de motifs fermés. On rappelle qu'un motif I de $\mathcal{D}$ est *fermé* si et seulement si, pour tout motif J tel que $I \subset J$, $\mathcal{S}upp(I, \mathcal{D}) > \mathcal{S}upp(J, \mathcal{D})$ est vérifiée. Soit I un motif fermé dans $\mathcal{D}$ et σ une symétrie $\mathcal{D}$. Comme σ est une symétrie, $\mathcal{S}upp(\sigma(I), \mathcal{D}) = \mathcal{S}upp(I, \mathcal{D})$ est vérifiée. Nous prouvons aussi que $\sigma(I)$ est aussi un motif fermé de $\mathcal{D}$. Soit I' un motif tel que $\sigma(I) \subset I'$ et $\mathcal{S}upp(\sigma(I), \mathcal{D}) \leq \mathcal{S}upp(I', \mathcal{D})$. On aura $I \subset \sigma^{-1}(I')$ et $\mathcal{S}upp(I, \mathcal{D}) \leq \mathcal{S}upp(\sigma^{-1}(I'), \mathcal{D})$. On obtient une contradiction, on en déduit que $\sigma(I)$ est fermé. Ainsi, notre approche d'élagage basé sur les symétries peut aussi être appliquée pour l'extraction des motifs fermés.

6 Symétries en fouille de données : état de l'art

En fouille de données, les symétries sont une propriété structurelle qui peut être exploitée pour élaguer l'espace de recherche ou réduire la taille de la sortie. Les symétries peuvent aussi, dans certaines applications, être vues comme étant une connaissance pertinente à extraire. Elles permettent, par exemple, aux analystes de données de mieux comprendre certaines corrélations entre les items. Beaucoup d'intérêt a été accordé à l'exploitation des symétries dans le domaine de l'intelligence artificielle, notamment pour les problèmes SAT (Benhamou et Saïs, 1994; Crawford et al., 1996) et CSP (Puget, 1993). En fouille de données les symétries sont principalement étudiées dans le cadre de la fouille de graphes (Vanetik, 2010; Desrosiers et al., 2007).

Dans (Desrosiers et al., 2007), les auteurs exploitent les symétries de sous graphes pour élaguer l'espace de recherche lors de la génération de sous graphes candidats. Les symétries (au sens automorphismes) sont utilisées dans (Vanetik, 2010) en tant que mesure d'intérêt. Dans, (Garrido, 2011), les auteurs proposent une approche pour détecter les symétries dans les réseaux sociaux.

Les symétries sont présentes aussi dans le clustering de données. Par exemple, (Murtagh et Contreras, 2010) traite le cas de clustering hiérarchique dans de grandes masses de données

pour détecter des symétries et d'autres motifs intéressants. Les symétries sont considérées dans ce cas comme une structure révélant des propriétés intrinsèques et invariantes dans les données.

Nous constatons enfin que peu de travaux se sont intéressés aux symétries dans le cadre de l'extraction de motifs ensemblistes. Nous mentionnons tout de même deux travaux traitant un cas particulier de symétries (Minato, 2006; Medina et al., 2005). En effet, les symétries traitées dans ces papiers, sont dites symétries de paires. Elles permettent d'échanger à chaque fois un couple d'items en laissant les autres items inchangés. (Minato, 2006) propose un algorithme ZBDD efficace permettant la détection des symétries de paires et discute les propriétés des items symétriques. Néanmoins, ce travail reste restreint à des cas très particuliers de symétries. Les paires d'items symétriques, appelés items clones par Medina et Nourine dans (Gély et al., 2005), sont utilisées pour expliquer pourquoi, dans certains cas, le nombre de règles d'une couverture minimale d'une relation est exponentiel en nombre d'items.

7 Conclusion et travaux futurs

Nous avons proposé dans cet article une approche d'élagage à base de symétries pour les problèmes d'extraction de motifs ensemblistes. Les symétries sont des propriétés structurelles qu'on détecte dans un grand nombre de bases de données. Elles ont déjà montré leur intérêt dans divers problèmes de satisfaction, d'optimisation et d'énumération. Dans cet article, nous avons montré comment une telle propriété peut être exploitée dans la phase d'élagage de l'algorithme pionnier d'extraction de motifs fréquents APRIORI permettant de réduire considérablement le nombre d'accès à la base et de calcul de fréquences. Nous avons montré expérimentalement que dans certains cas, les calculs de fréquences sont réduits d'environ 24% grâce à l'exploitation des symétries. L'un des points forts de cette approche réside dans le fait que la détection des symétries ne s'effectue qu'une et une seule fois pour chaque jeu de données et est indépendante de toute tâche d'extraction de motifs. Nous avons montré aussi que notre approche peut être généralisée à d'autres tâches d'extraction de motifs tant que le prédicat de sélection en question est stable par symétrie.

Dans nos futurs travaux, nous envisageons d'étendre notre approche d'élagage à base de symétries à la fouille de séquences. D'autres problèmes de fouille de données tel que le clustering pourrait aussi tirer profit de la présence des symétries dans les données. Nous souhaitons aussi trouver un moyen d'exploiter les symétries locales dans les données (symétries dans une portion des données uniquement).

Références

Agrawal, R. et R. Srikant (1994). Fast algorithms for mining association rules in large databases. In *Proceedings of VLDB'94*, pp. 487–499.

Aloul, F. A., A. Ramani, I. L. Markov, et K. A. Sakallah (2003). Solving difficult instances of boolean satisfiability in the presence of symmetry. *Transactions on Computer Aided Design*.

Benhamou, B. et L. Saïs (1994). Tractability through symmetries in propositional calculus. *Journal of Automated Reasoning 12*(1), 89–102.

Crawford, J., M. L. Ginsberg, E. Luck, et A. Roy (1996). Symmetry-breaking predicates for search problems. In *KR'96*, pp. 148–159.

Desrosiers, C., P. Galinier, P. Hansen, et A. Hertz (2007). Improving frequent subgraph mining in the presence of symmetry. In *Proceedings of MLG'2007*.

Garrido, A. (2011). Symmetry in complex networks. *Symmetry 3*(1), 1–15.

Gély, A., R. Medina, L. Nourine, et Y. Renaud (2005). Uncovering and reducing hidden combinatorics in guigues-duquenne bases. In *ICFCA'05*, pp. 235–248.

Gent, I. P. et B. Smith (2000). Symmetry breaking in constraint programming. In *Proceedings of the Thirteenth European Conference on Artificial Intelligence(ECAI'00)*, pp. 599–603.

Jabbour, S., L. Sais, Y. Salhi, et K. Tabia (2012). Symmetries in itemset mining. In *20th European Conference on Artificial Intelligence (ECAI'2012)*, pp. 432–437.

Junttila, T. A. et P. Kaski (2007). Engineering an efficient canonical labeling tool for large and sparse graphs. In *ALENEX*.

Liberti, L. (2012). Reformulations in mathematical programming : automatic symmetry detection and exploitation. *Math. Program. 131*(1-2), 273–304.

Mannila, H. et H. Toivonen (1997). Levelwise search and borders of theories in knowledge discovery. *Data Min. Knowl. Discov. 1*(3), 241–258.

Margot, F. (2003). Exploiting orbits in symmetric ilp. *Math. Program. 98*(1-3), 3–21.

McKay, B. D. (1981). Practical graph isomorphism. *Congressus Numerantium* (30), 47–87.

Medina, R., C. Noyer, et O. Raynaud (2005). Efficient algorithms for clone items detection. In *CLA'05*, pp. 70–81.

Minato, S.-i. (2006). Symmetric item set mining based on zero-suppressed bdds. In *DS'06*, pp. 321–326.

Murtagh, F. et P. Contreras (2010). Hierarchical clustering for finding symmetries and other patterns in massive, high dimensional datasets. *CoRR abs/1005.2638*.

Puget, J.-F. (1993). On the satisfiability of symmetrical constrained satisfaction problems. In *ISMIS*, pp. 350–361.

Vanetik, N. (2010). Mining graphs with constraints on symmetry and diameter. In *International Workshops on Web-Age Information Management - WAIM 2010*, pp. 1–12.

Zheng, Z., R. Kohavi, et L. Mason (2001). Real world performance of association rule algorithms. In *Proceedings of KDD '01*, pp. 401–406. ACM.

Summary

In this paper, we show how symmetries, a fundamental structural property, can be used to prune the search space in itemset mining problems. Our approach is based on a dynamic integration of symmetries in APRIORI-like algorithms to prune the set of possible candidate patterns. More precisely, for a given itemset, symmetry can be applied to deduce other itemsets while preserving their properties. We also show that our symmetry-based pruning approach can be extended to the general Mannila and Toivonen pattern mining framework. Experimental results highlight the usefulness and the efficiency of our symmetry-based pruning approach.