Extraire les motifs minimaux efficacement et en profondeur

Arnaud Soulet*, François Rioult**

*Université François-Rabelais de Tours, LI EA 6300,
Campus de Blois, 41000 Blois
arnaud.soulet@univ-tours.fr
**Université de Caen Basse-Normandie, GREYC UMR 6072,
Campus II Côte de Nacre, 14000 Caen
francois.rioult@unicaen.fr

Résumé. Les représentations condensées ont fait l'objet de nombreux travaux depuis 15 ans. Tandis que les motifs maximaux des classes d'équivalence ont reçu beaucoup d'attention, les motifs minimaux sont restés dans l'ombre notamment à cause de la difficulté de leur extraction. Dans ce papier, nous présentons un cadre générique concernant l'extraction de motifs minimaux en introduisant la notion de système minimisable d'ensembles. Il permet de considérer des langages variés comme les motifs ensemblistes ou les chaînes de caractères, mais aussi différentes métriques dont la fréquence. Ensuite, pour n'importe quel système minimisable d'ensembles, nous introduisons un test de minimalité rapide permettant d'extraire en profondeur les motifs minimaux. Nous démontrons que l'algorithme proposé est *polynomial-delay* et *polynomial-space*. Des expérimentations sur les benchmarks traditionnels complètent notre étude.

1 Introduction

La minimalité est un concept essentiel de l'extraction de motifs. Pour une fonction f et un langage $\mathcal{L}$, un motif X est minimal si sa valeur pour f est distincte de celle de chacun de ses sous-ensembles. La collection de tous les motifs minimaux constitue une représentation condensée de $\mathcal{L}$ adéquate à f : il est possible de retrouver $f(Y)$ pour n'importe quel motif $Y \in \mathcal{L}$. Typiquement, l'ensemble des itemsets libres (Boulicaut et al., 2000) est une représentation condensée de tous les itemsets (dans ce cas, f et $\mathcal{L}$ sont respectivement la fréquence et le langage des itemsets). Bien sûr, il est souvent plus efficace d'extraire les motifs minimaux plutôt que la totalité des motifs. En plus, les motifs minimaux ont de nombreuses applications utiles notamment en extraction de connaissances : la production de bases de règles d'association, la construction de classifieurs ou la génération des traverses minimales. La minimalité a été étudiée dans le cas de différentes fonctions (comme la fréquence (Calders et al., 2004) ou les fonctions condensables (Soulet et Crémilleux, 2008)) et avec des langages variés (dont les itemsets (Boulicaut et al., 2000) et les séquences (Lo et al., 2008)). Bien que la minimalité ait des avantages évidents (Li et al., 2006), elle reste peu explorée en comparaison de la maximalité (i.e., les motifs fermés). En particulier, à notre connaissance, il n'existe pas de cadre suffisamment général comme celui proposé par Arimura et Uno (2009) pour les maximaux.

Nous pensons qu'un des inconvénients importants des motifs minimaux réside dans leur difficile extraction. L'efficacité limitée des algorithmes existants découle principalement de leur approche par niveaux (Boulicaut et al., 2000; Soulet et Crémilleux, 2008; Casali et al., 2005) (i.e., en largeur avec la méthode générer-et-tester). Comme ils stockent tous les candidats d'un même niveau en mémoire durant la phase de génération, l'extraction peut échouer par manque de mémoire. Pour éviter cet écueil, il semble préférable d'adopter un parcours en profondeur qui souvent consomme moins de mémoire tout en restant très rapide. Cependant, vérifier si la minimalité est satisfaite ou non est vraiment difficile avec un parcours en profondeur. Dans le cas de la fréquence et des itemsets, la meilleure façon d'évaluer la minimalité d'un motif (disons abc) est de comparer sa fréquence avec celle de ses généralisations directes (ici, ab, ac et bc). Mais, quand le parcours en profondeur atteint le motif abc, seules les fréquences de a et ab ont été précédemment calculées. Comme les fréquences de ac et bc ne sont pas connues, il est impossible de vérifier si la fréquence de abc est bien strictement inférieure à celle de ac et bc. Pour résoudre ce problème, Calders et Goethals (2005); Liu et al. (2008); Szathmary et al. (2009) ont adopté un parcours en profondeur en énumérant les items dans l'ordre inverse. Par exemple, quand l'itemset abc est atteint par ce nouveau parcours, c, b, bc, a, ac et bc ont précédemment été parcourus et leurs fréquences sont donc connues pour vérifier si abc est minimal. Malheureusement, une telle méthode requièrt de stocker de nombreux motifs en mémoire (ici, c, b, bc, etc) en utilisant soit un *trie* pour Calders et Goethals (2005) soit une table de hachage pour Liu et al. (2008); Szathmary et al. (2009). Pour cette raison, les propositions existantes en profondeur faites par Calders et Goethals (2005); Liu et al. (2008); Szathmary et al. (2009) ne résolvent pas le problème de consommation mémoire.

Contributions. L'objectif principal de cet article est de présenter un cadre générique et efficace pour l'extraction des motifs minimaux, et ce en proposant un algorithme en profondeur peu consommateur en mémoire. Nous introduisons la notion de *système minimisable d'ensembles* qui est au coeur de la définition de notre cadre. Ce dernier couvre un très large spectre de motifs minimaux incluant tous les langages et les mesures étudiés par Soulet et Crémilleux (2008); Arimura et Uno (2009). Une vérification rapide de la minimalité lors d'un parcours en profondeur est réalisée grâce à la notion d'*objets critiques* dont la définition découle du système minimisable d'ensembles considéré. En s'appuyant sur cette nouvelle technique, nous proposons l'algorithme DEFME. Il extrait tous les motifs minimaux d'un système minimisable d'ensembles en utilisant un parcours en profondeur. À notre connaissance, il s'agit du premier algorithme qui énumère tous les motifs minimaux en *polynomial delay* et avec une consommation linéaire d'espace par rapport au jeu de données initial.

2 Système d'ensembles pour les motifs minimaux

2.1 Définitions

Un *système d'ensembles* $(\mathcal{F}, E)$ est une collection $\mathcal{F}$ de sous-ensembles d'un ensemble E (i.e. $\mathcal{F}$ est un sous-ensemble des parties de E). Un membre de $\mathcal{F}$ est appelé un *ensemble acceptable*. Un système d'ensembles *fortement accessible* $(\mathcal{F}, E)$ est un système d'ensembles où pour tous les ensembles acceptables X, Y satisfaisant $X \subset Y$, il existe un élément $e \in Y \setminus X$ tel que $Xe \in \mathcal{F}$ [1]. Bien entendu, les itemsets rentrent dans ce cadre avec le système

1. Nous utilisons la notation Xe pour dénoter $X \cup \{e\}$.

d'ensembles $(2^{\mathcal{I}}, \mathcal{I})$ où $\mathcal{I}$ est l'ensemble des items. $(2^{\mathcal{I}}, \mathcal{I})$ est même fortement accessible. Mais la notion de système d'ensembles permet également de considérer des langages plus sophistiqués. Par exemple, il est facile de construire un ensemble $\mathcal{F}_S$ dénotant la collection des sous-chaînes de $S = abracadabra$ en encodant chaque sous-chaîne $s_{k+1}s_{k+2}\dots s_{k+n}$ par un ensemble $\{(s_{k+1}, 1), (s_{k+2}, 2), \dots, (s_{k+n}, n)\}$. Le système d'ensembles $(\mathcal{F}_S, E_S = \bigcup \mathcal{F}_S)$ est aussi fortement accessible. Les systèmes d'ensembles ont déjà été utilisés à plusieurs reprises pour décrire des problèmes d'extraction de motifs (Arimura et Uno, 2009).

Intuitivement, un motif décrit toujours un ensemble d'objets. Pour un motif donné, cet ensemble d'objets peut être obtenu par le biais d'un *opérateur de couverture* formalisé ainsi :

Définition 1 (Opérateur de couverture) *Soit un ensemble d'objets $\mathcal{O}$, un opérateur de couverture $cov : 2^E \to 2^{\mathcal{O}}$ satisfait $cov(X \cup Y) = cov(X) \cap cov(Y)$ pour tout $X, Y \in 2^E$.*

Cette définition indique que la couverture de l'union de deux motifs correspond exactement à l'intersection de leurs deux couvertures. Pour les itemsets, un opérateur de couverture naturel est l'extension d'un itemset X qui retourne tous les identifiants des tuples contenant X : $cov_{\mathcal{I}}(X) = \{o \in \mathcal{O} \mid X \subseteq o\}$. La couverture permet surtout de dériver des informations utiles : la cardinalité de $cov_{\mathcal{I}}(X)$ correspond à la fréquence de X. Dans le contexte des chaînes, la liste des index d'une chaîne X définit également un opérateur de couverture : $cov_S(X) = \{i \mid \forall (s_j, j) \in X,\ (s_j, j+i) \in S\}$. Continuons notre exemple sur la chaîne $S = abracadabra$ avec quelques listes d'index $cov_S(\{(a,1)\}) = \{0, 3, 5, 7, 10\}$ et $cov_S(\{(b,2)\}) = \{0, 7\}$. On vérifie alors que $cov_S(\{(a,1),(b,2)\}) = cov_S(\{(a,1)\}) \cap cov_S(\{(b,2)\}) = \{0, 7\}$.

Pour certains langages, le même motif est décrit par plusieurs ensembles distincts et alors, il est nécessaire d'avoir une forme canonique (pour éviter de l'extraire plusieurs fois, par exemple). Typiquement, considérons l'ensemble $\{(a,1),(b,2),(r,3)\}$ correspondant à la chaîne abr. Son suffixe $\{(b,2),(r,3)\}$ désigne la même chaîne br que $\{(b,1),(r,2)\}$. Dans notre cas, $\{(b,1),(r,2)\}$ sera la forme canonique de br car le premier élément commence par 1. Pour retrouver la forme canonique d'un motif, nous introduisons un nouvel opérateur :

Définition 2 (Opérateur canonique) *Etant donnés deux systèmes d'ensembles $(\mathcal{F}, E)$ et $(\mathcal{G}, E)$, un opérateur canonique $\phi : \mathcal{F} \cup \mathcal{G} \to \mathcal{F}$ est une fonction satisfaisant (i) $X \subset Y \Rightarrow \phi(X) \subset \phi(Y)$ et (ii) $X \in \mathcal{F} \Rightarrow \phi(X) = X$ pour tous les ensembles $X, Y \in \mathcal{G}$.*

Dans cette définition, la propriété (i) nous garantit que les formes canoniques de deux ensembles comparables (au sens de l'inclusion) restent comparables. La propriété (ii) signifie que le système d'ensembles $(\mathcal{F}, E)$ inclut toutes les formes canoniques. Continuons encore notre exemple sur les chaînes : on peut constater que $\phi_S : \{(s_k, k), (s_{k+1}, k+1), \dots, (s_{k+n}, n)\} \mapsto \{(s_k, 1), (s_{k+1}, 2), \dots, (s_{k+n}, n-k+1)\}$ satisfait les deux propriétés désirées (i) et (ii). Par exemple, $\phi_S(\{(b,2),(r,3)\})$ retourne la forme canonique de $\{(b,2),(r,3)\}$: $\{(b,1),(r,2)\}$.

2.2 Système minimisable d'ensembles

Plutôt que de considérer un système d'ensembles dans son intégralité, il peut s'avérer judicieux d'en sélectionner une partie qui apporte la même quantité d'information (au sens de l'opérateur de couverture). Pour cela, il est nécessaire que ce système d'ensembles et l'opérateur de couverture visé forment un système *minimisable* d'ensembles :

Définition 3 (Système minimisable d'ensembles) *Un système minimisable d'ensembles est un tuple $\langle(\mathcal{F}, E), \mathcal{G}, cov, \phi\rangle$ où :*

- *$(\mathcal{F}, E)$ est un système d'ensembles fini et fortement accessible. Un ensemble acceptable dans $\mathcal{F}$ est appelé motif.*
- *$(\mathcal{G}, E)$ est un système d'ensembles fini et fortement accessible satisfaisant pour chaque couple d'ensembles acceptables $X, Y \in \mathcal{F}$ tels que $X \subseteq Y$ et chaque élément $e \in E$, $X \backslash \{e\} \in \mathcal{G} \Rightarrow Y \backslash \{e\} \in \mathcal{G}$. Un ensemble acceptable de $\mathcal{G}$ est appelé une généralisation.*
- *$cov : 2^E \rightarrow 2^{\mathcal{O}}$ est un opérateur de couverture.*
- *$\phi : \mathcal{F} \cup \mathcal{G} \rightarrow \mathcal{F}$ est un opérateur canonique tel que pour chaque ensemble acceptable $X \in \mathcal{G}$, on ait $cov(\phi(X)) = cov(X)$.*

Illustrons maintenant le rôle de $\mathcal{G}$ par rapport à $\mathcal{F}$ dans le cas des chaînes. En fait, $\mathcal{G}_S$ regroupe tous les suffixes de n'importe quel motif de $\mathcal{F}_S$. Typiquement, $\{(b,2),(r,3)\} \in \mathcal{G}_S$ est une généralisation de $\{(a,1),(b,2),(r,3)\} \in \mathcal{F}_S$. Comme dit au-dessus, $\{(b,2),(r,3)\}$ a une forme équivalente dans $\mathcal{F}_S$: $\phi_S(\{(b,2),(r,3)\}) = \{(b,1),(r,2)\}$. Par convention, nous étendons la définition de cov_S et $\mathcal{G}_S$ en considérant que $cov_S(\phi_S(X)) = cov_S(X)$. De plus, on voit que $\mathcal{G}_S$ satisfait la propriété désirée par rapport à $\mathcal{F}_S$: pour tous les ensembles acceptables $X, Y \in \mathcal{F}_S$ tels que $X \subseteq Y$ et chaque élément $e \in E_S$, $X \setminus \{e\} \in \mathcal{G}_S \Rightarrow Y \setminus \{e\} \in \mathcal{G}_S$. En effet, si $X \setminus \{e\}$ est un suffixe de X, cela signifie que e est la première lettre. Si nous considérons une spécialisation de X et que nous retirons la première lettre, nous obtenons aussi un suffixe qui appartient à $\mathcal{G}_S$. Par conséquent, $\langle(\mathcal{F}_S, E_S), \mathcal{G}_S, cov_S, \phi_S\rangle$ est minimisable.

Bien évidemment, un système minimisable d'ensembles peut être réduit à un système de cardinalité inférieure dont les motifs sont appelés *motifs minimaux* :

Définition 4 (Motif minimal) *Un motif X est minimal pour $\langle(\mathcal{F}, E), \mathcal{G}, cov, \phi\rangle$ ssi $X \in \mathcal{F}$ et pour chaque généralisation $Y \in \mathcal{G}$ telle que $Y \subset X$, $cov(Y) \neq cov(X)$. $\mathcal{M}(\mathcal{S})$ dénote l'ensemble des motifs minimaux.*

La définition 4 signifie qu'un motif est minimal dès que sa couverture diffère de celle de toutes ses généralisations. Par exemple, pour l'opérateur de couverture cov_S, les motifs minimaux ont une couverture *strictement* plus petite que celles de leurs généralisations. La chaîne ab n'est pas minimale à cause de son suffixe b car $cov_S(\{(b,2))\}) = cov_S(\{(a,1),(b,2)\}) = \{0,7\}$. Dans notre exemple, la collection des chaînes minimales est $\mathcal{M}(\mathcal{S}_S) = \{a, b, r, c, d, ca, ra, da\}$.

Etant donné un système minimisable d'ensembles $\mathcal{S} = \langle(\mathcal{F}, E), \mathcal{G}, cov, \phi\rangle$, l'extraction des motifs minimaux consiste à énumérer tous les motifs minimaux de $\mathcal{S}$.

3 Énumération des motifs minimaux

Le but de cette section est de proposer une methode d'extraction en profondeur des motifs minimaux (section 3.3). Pour cela, nous nous appuyons sur deux idées principales : l'élagage de l'espace de recherche (section 3.1) et la vérification rapide de la minimalité (section 3.2).

Auparavant il est important de rappeler que les motifs minimaux suffisent pour déduire la couverture de tout motif. Nous considérons désormais un système minimisable d'ensembles $\mathcal{S} = \langle(\mathcal{F}, E), \mathcal{G}, cov, \phi\rangle$. Les motifs minimaux $\mathcal{M}(\mathcal{S})$ consituent une représentation sans perte de tous les motifs de $\mathcal{F}$:

Théorème 1 (Représentation condensée) *L'ensemble des motifs minimaux est une représentation condensée de $\mathcal{F}$ adéquate pour le calcul de cov : pour tout motif $X \in \mathcal{F}$, il existe $Y \subseteq X$ tel que $\phi(Y) \in \mathcal{M}(\mathcal{S})$ et $cov(\phi(Y)) = cov(X)$.*

Le théorème 1 signifie que la couverture de tout motif de $\mathcal{S}$ peut être déduite de $\mathcal{M}(\mathcal{S})$ (les preuves sont omises par manque de place). Par exemple, la couverture du motif non minimal $\{(a,1),(b,2)\}$ est égale à celle du motif minimal $\phi_S(\{(b,2)\}) = \{(b,1)\} : cov_S(\{(a,1),(b,2)\}) = cov_S(\{(b,1)\}) = \{0,7\}$. Il est préférable d'extraire $\mathcal{M}(\mathcal{S})$ plutôt que $\mathcal{S}$ car sa taille est plus petite que celle du système complet.

3.1 Élagage de l'espace de recherche

Le premier problème auquel nous sommes confrontés est plutôt classique. Étant donné un système minimisable d'ensembles $\mathcal{S} = \langle (\mathcal{F}, E), \mathcal{G}, cov, \phi \rangle$, le nombre de motifs $|\mathcal{F}|$ est en général très grand (dans le pire des cas, il atteint $2^{|E|}$). Il est donc absolument nécessaire de ne pas parcourir l'espace de recherche exhaustivement, mais de se concentrer sur les motifs minimaux. Heureusement, des techniques efficaces peuvent être utilisées pour élaguer l'espace de recherche, grâce à la propriété d'anti-monotonie de la contrainte de minimalité :

Théorème 2 (Système indépendant) *Si un motif X est minimal pour $\mathcal{S}$, alors tout motif $Y \in \mathcal{F}$ satisfaisant $Y \subseteq X$ est aussi minimal pour $\mathcal{S}$.*

La preuve de ce théorème est fortement dépendante d'un lemme clé indiquant qu'un motif non minimal a une généralisation directe possédant la même couverture.

Lemme 1 *Si X n'est pas mininal, $\exists e \in X$ tel que $X \setminus \{e\} \in \mathcal{G}$ et $cov(X) = cov(X \setminus \{e\})$.*

Par exemple, comme la chaîne da est minimale, les sous-chaînes d et a le sont également. En outre, comme ab n'est pas minimal, la chaîne abr ne l'est pas non plus. Cela signifie que la chaîne ab est un point de rupture dans l'espace de recherche. En pratique, l'élagage par anti-monotonie est reconnu comme un outil puissant, quelque soit le type de parcours de l'espace de recherche, par niveau ou en profondeur.

3.2 Vérification rapide de la minimalité

La difficulté principale de l'extraction des motifs minimaux est de tester si un motif est minimal ou non. Comme mentionné précédemment, ceci est particulièrement difficile lors d'un parcours en profondeur, car tous les sous-ensembles du motif à tester n'ont pas été énumérés. En effet, les approches en profondeur n'ont accès qu'à la branche parente, au contraire des approches par niveau. Pour résoudre ce problème, nous introduisons le concept d'*objets critiques* inspiré de celui d'hyper-arête critique, introduit pour le calcul des traverses minimales par Murakami et Uno (2013). Intuitivement, les objets critiques d'un élément e pour un motif X sont les objets qui ne sont pas couverts par X du fait de e.

Nous donnons maintenant une définition formelle des objets critiques, dérivant d'un opérateur de couverture quelconque :

Définition 5 (Objets critiques) *Pour un motif X, les objets critiques d'un élément $e \in X$, notés $\widehat{cov}(X, e)$ sont l'ensemble d'objets qui appartiennent à la couverture de X privé de e mais pas à la couverture de e : $\widehat{cov}(X, e) = cov(X \setminus e) \setminus cov(e)$.*

Illustrons le concept d'objets critiques sur notre exemple. Pour $\{(a, 1), (b, 2)\}$, les objets critiques $\widehat{cov}_S(ab, a)$ de l'élément $(a, 1)$ correspondent à $\emptyset$ $(= \{0, 7\} \setminus \{0, 3, 5, 7, 10\})$. Cela signifie que l'ajout de a à b n'a pas d'impact sur la couverture de ab. En revanche, pour le même motif, les objets critiques de $(b, 2)$ sont $\{3, 5, 10\}$ $(= \{0, 3, 5, 7, 10\} \setminus \{0, 7\})$. C'est à cause de l'élément b que ab ne couvre pas les objets $\{3, 5, 10\}$.

Les objets critiques sont centraux pour notre proposition : 1) les objets critiques caractérisent facilement les motifs minimaux ; et 2) les objets critiques peuvent être calculés efficacement au sein d'un algorithme en profondeur.

Caractérisation de la minimalité La réciproque du lemme 1 indique qu'un motif est minimal si sa couverture diffère de celle d'une de ses généralisations. Nous pouvons reformuler cette définition grâce à la notion d'objets critiques :

Propriété 1 (Minimalité) *$X \in \mathcal{F}$ est minimal si $\forall e \in X$ tel que $X \setminus e \in \mathcal{G}$, $\widehat{cov}(X, e) \neq \emptyset$.*

Typiquement, comme b est une généralisation de la chaîne ab, et $\widehat{cov}_S(ab, a)$ est vide, ab n'est pas minimal. La propriété 1 signifie que tester si un candidat X est minimal ne nécessite que la connaissance des objets critiques relativement aux éléments de X. À la différence de la définition traditionnelle, il n'est pas nécessaire de disposer d'informations sur les sous-ensembles. Les objets critiques nous permettent donc de concevoir un algorithme en profondeur si (et seulement si) le calcul des objets critiques ne requiert pas non plus d'informations sur les sous-ensembles.

Calcul efficace d'objets critiques Selon un parcours en profondeur, nous voulons mettre à jour les objets critiques d'un élément e pour le motif X lorsqu'un nouvel élément e' est ajouté à X. Nous montrons que, dans ce cas, les objets critiques peuvent être calculés efficacement par intersection des objets critiques $\widehat{cov}(X, e)$ avec la couverture du nouvel élément e' :

Propriété 2 *L'égalite suivante est vraie pour tout motif $X \in \mathcal{F}$ et deux élément $e, e' \in E$:*

$$\widehat{cov}(Xe', e) = \widehat{cov}(X, e) \cap cov(e').$$

Par exemple, la définition 5 donne $\widehat{cov}_S(a, a) = \{1, 2, 4, 6, 8, 9\}$. Comme $cov_S(b) = \{0, 7\}$, on obtient que $\widehat{cov}_S(ab, a) = \widehat{cov}_S(a, a) \cap cov_S(b) = \{1, 2, 4, 6, 8, 9\} \cap \{0, 7\} = \emptyset$. Il est intéressant de noter que la propriété 2 nous permet de calculer les objets critiques de tout élément d'un motif X en ne disposant que de l'information sur une seule branche du parcours. C'est une situation idéale pour la conception d'un algorithme en profondeur.

3.3 Algorithme DEFME

L'algorithme DEFME prend en entrée le motif courant X et la queue $tail$ des éléments restant à vérifier, il retourne tous les motifs minimaux contenant X, basés sur $tail$. Plus précisément, la ligne 1 teste si X est minimal ou non. Si X est minimal, il est affiché (ligne 2). Les lignes 3 à 14 explorent le sous-arbre contenant X selon la queue. Pour chaque élément e

pour lequel Xe est un motif de $\mathcal{F}$ (ligne 4) (propriété 1), la branche est construite à l'aide des informations nécessaires. La ligne 7 met à jour la couverture et les lignes 8 à 11 calculent les objets critiques en utilisant la propriété 2. Enfin, la fonction DEFME est appelée récursivement à la ligne 12 avec la queue mise à jour à la ligne 5.

Algorithm 1 DEFME$(X, tail)$

Input: X est un motif $tail$ est l'ensemble des items restant à utiliser pour générer les candidats. Valeurs initiales : $X = \emptyset$, $tail = E$.
Output: calcule les motifs minimaux de manière incrémentale polynomiale.

```
1: if ∀e ∈ X, c̃ov(X, e) ≠ ∅ then
2:    print X
3:    for all e ∈ tail do
4:       if Xe ∈ F then
5:          tail := tail \ {e}
6:          Y := Xe
7:          cov(Y) := cov(X) ∩ cov(e)
8:          c̃ov(Y, e) := cov(X) \ cov(e)
9:          for all e' ∈ X do
10:            c̃ov(Y, e') := c̃ov(X, e') ∩ cov(e)
11:         end for
12:         DEFME(Y, tail)
13:      end if
14:   end for
15: end if
```

Les théorèmes 3 et 4 montrent que l'algorithme DEFME possède un comportement efficace, tant en terme d'espace que de temps de calcul. Cette efficacité découle du calcul économique des couvertures et des objets critiques, ainsi que la propriété suivante l'explique :

Propriété 3 *Pour tout motif $X \in \mathcal{F}$, on a $|cov(X)| + \sum_{e \in X} |\widetilde{cov}(X, e)| \leq |cov(\emptyset)|$.*

La propriété 3 signifie que, pour un motif, le stockage de sa couverture plus celui de ses objets critiques est majoré par le nombre total d'objets (i.e., $|cov(\emptyset)|$). La déduction de l'espace de mémoire nécessaire pour l'algorithme est donc directe :

Théorème 3 (Complexité polynomiale en espace) *$\mathcal{M}(\mathcal{S})$ est énumerable en espace $O(|cov(\emptyset)| \cdot m)$ où m est la longueur maximale d'une solution de $\mathcal{F}$.*

En pratique, l'espace mémoire utilisé par l'algorithme est très limité, car m est petit. En outre, le delai entre deux motifs est polynomial :

Théorème 4 (Complexité polynomiale en délai) *$\mathcal{M}(\mathcal{S})$ est énumerable en temps $O(|E|^2 \cdot |cov(\emptyset)|)$ par motif minimal.*

DEFME requiert un nombre polynomial d'opérations entre deux motifs, en supposant que l'oracle d'appartenance agit en temps polynomial (ligne 4). En effet, le calcul de la couverture et celui des objets critiques (lignes 7 à 11) est linéaire avec le nombre d'objets, grâce à la propriété 3 ; la boucle à la ligne 3 n'excède pas $|E|$ itérations et finalement, le nombre de retours en arrière consécutifs est au plus de $|E|$.

4 Étude expérimentale

Le but de nos expériences est de quantifier le bénéfice apporté par DEFME, tant sur le plan de l'efficacité que de la concision. Nous montrons son efficacité pour l'extraction de motifs

libres, pour laquelle plusieurs prototypes existent dans la littérature. Ensuite, nous utilisons DEFME pour extraire la collection des chaînes minimales et nous comparons sa taille avec celle des chaînes maximales. Tous les tests ont été effectués sur une machine Linux dotée d'un processeur Opteron à 2,2 GHz et de 200 Go de RAM.

4.1 Extraction des motifs libres

Nous avons réalisé un prototype de DEFME pour l'extraction des motifs ensemblistes, en tant que preuve de concept, et nous l'avons comparé avec deux autres prototypes : ACMINER, basé sur un algorithme par niveaux (Boulicaut et al., 2000) et NDI [2] utilisant un parcours en profondeur réordonnant les items (Calders et Goethals, 2005). Nous avons pour cela réalisé des expérimentations sur les données de test du FIMI et du Discovery Challenge de PKDD 2004 [3]. Les trois premières colonnes de la table 1 donnent les caractéristiques des jeux de données. Les trois colonnes suivantes indiquent les temps d'exécution et les trois dernières indiquent la consommation de mémoire.

				temps (s)			mémoire (ko)		
données	objets	items	minsup	ACMINER	NDI	DEFME	ACMINER	NDI	DEFME
74x822	74	822	88%	échec	échec	**45**	échec	échec	**3 328**
90x27679	90	27 679	91%	échec	échec	**79**	échec	échec	**13 352**
chess	3 196	75	22%	6 623	**187**	192	3 914 588	1 684 540	**8 744**
connect	67 557	129	7%	34 943	**115**	4 873	2 087 216	1 181 296	**174 680**
pumsb	49 046	2 113	51%	70 014	**212**	548	7 236 812	1 818 500	**118 240**
pumsb*	49 046	2 088	5%	21267	**202**	4600	5 175 752	2 523 384	**170 632**

TAB. 1 – *Caractéristiques des données de test et résultats pour l'extraction des motifs libres.*

Les meilleurs performances sont indiquées par l'utilisation d'une police grasse dans la table 1 pour les temps d'exécution et la consommation mémoire. ACMINER est de loin le prototype le plus lent. Son approche par niveau est particulièrement pénalisée par l'importance de la consommation mémoire. Excepté sur les données génomiques et sur chess, les temps d'exécution de NDI surpassent clairement ceux de DEFME. À titre d'information, la figure 1, à gauche, indique la vitesse de DEFME, selon différents seuils de support minimal. Cette figure représente le nombre de motifs minimaux calculés par seconde.

Concernant la consommation mémoire, DEFME est (comme prévu) le plus efficace des algorithmes. Dans certains cas, augmenter la quantité de mémoire disponible ne suffirait pas à traiter les données les plus difficiles. Ici, ACMINER et NDI ne peuvent traiter les données génomique même avec 200 Go de mémoire, à des seuils de support minimal pourtant élévés. Plus précisément, la figure 1, à droite, représente le rapport entre les utilisations de mémoire de NDI et de DEFME, selon le seuil de support. On remarque grâce à ce rapport que NDI explose en mémoire. Rappelons que DEFME travaille en mémoire bornée et n'est donc pas limité lors de faibles seuils de support.

4.2 Extraction de chaînes minimales

Dans cette section, nous adoptons le formalisme des chaînes, utilisé dans notre exemple récurrent. Nous avons comparé notre algorithme d'extraction de chaînes minimales avec le

2. Ce prototype permet d'extraire les motifs libres en fixant le paramètre de profondeur à 1.
3. `fimi.ua.ac.be/data/` et `lisp.vse.cz/challenge/ecmlpkdd2004/`

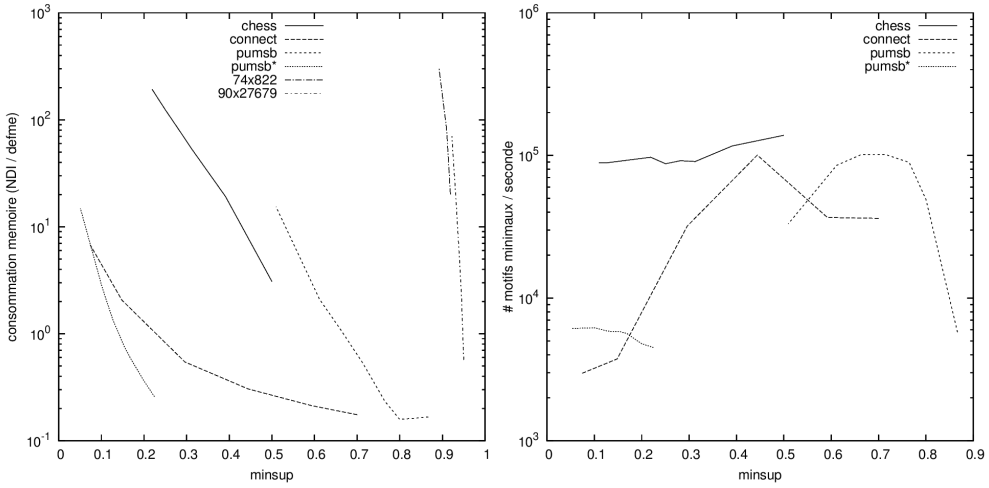

FIG. 1 – *Ratio de la vitesse d'extraction (à gauche) et de la consommation mémoire (à droite) de* NDI *par* DEFME.

prototype MAXMOTIF fourni par Takeaki Uno, extrayant les motifs minimaux. Notre but est de comparer la taille des représentations condensées fondées sur les chaînes minimales avec celles utilisant les chaînes maximales. Nous n'avons pas reporté les temps d'exécution car MAXMOTIF, développé en Java, est bien plus lent que DEFME. Les expérimentations ont été effectuées sur chromosom [4] et msnbc issu du dépôt UCI pour l'apprentissage automatique [5].

La figure 2 reporte les quantités de chaînes, de chaînes minimales et de chaînes maximales extraites dans chromosom (à gauche) et msnbc (à droite), selon le seuil de support minimal. Le nombre de ces chaînes augmente bien-sûr lorsque ce seuil diminue. On note que les deux représentations condensées des minimaux et des clos deviennent particulièrement intéressantes pour de faibles seuils de support. Le nombre de chaînes minimales est clairement plus grand que celui de chaînes maximales, mais l'écart n'est pas aussi important que dans le cas des motifs libres ou fermés. À ces seuils de supports, l'écart des moyennes des longueurs entre minimaux et maximaux est d'un item pour chromosom et de trois pour msnbc.

5 Travaux relatifs

La collection des motifs minimaux est une forme de représentation condensée. Un grand nombre de représentations condensées ont été proposées dans la littérature (Calders et al., 2004; Hamrouni, 2012) : les motifs fermés (Pasquier et al., 1999), les motifs libres (Boulicaut et al., 2000), les motifs essentiels (Casali et al., 2005), les motifs non-dérivables (Calders et Goethals, 2005), etc. Les deux idées fondatrices des représentations condensées sont les classes d'équivalences souvent issues d'un opérateur de fermeture (Hamrouni, 2012) et le principe d'inclusion-exclusion. Comme ce principe ne fonctionne que pour la fréquence, cet article

4. Ces données accompagnent MAXMOTIF : `research.nii.ac.jp/~uno/codes.htm`
5. `www.ics.uci.edu/~mlearn`

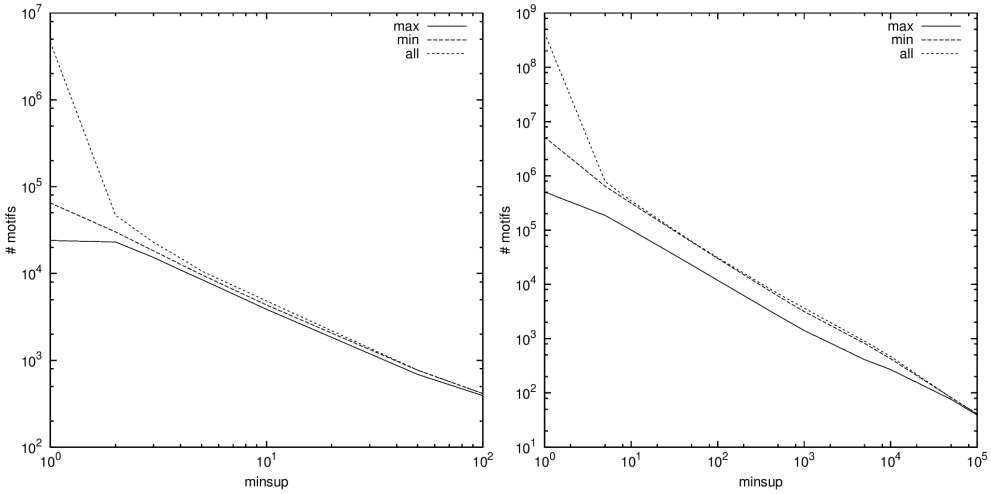

FIG. 2 – *Nombre de chaînes dans* chromosom *(à gauche) et dans* msnbc *(à droite).*

s'est focalisé uniquement sur les motifs minimaux en considérant les classes d'équivalence. En particulier, comme indiqué aupravant le système $\mathcal{S}_{\mathcal{I}} = \langle (2^{\mathcal{I}}, \mathcal{I}), 2^{\mathcal{I}}, cov_{\mathcal{I}}, Id \rangle$ est minimisable et $\mathcal{M}(\mathcal{S}_{\mathcal{I}})$ correspond exactement aux itemsets libres. La fréquence de chaque itemset est calculée en utilisant la cardinalité de l'opérateur de couverture. Remplacer cet opérateur de couverture avec $\overline{cov_{\mathcal{I}}} : X \mapsto \{o \in \mathcal{O} \mid X \cap o = \emptyset\}$ conduit à un nouveau système minisable d'ensembles $\langle (2^{\mathcal{I}}, \mathcal{I}), 2^{\mathcal{I}}, \overline{cov_{\mathcal{I}}}, Id \rangle$ dont les motifs minimaux sont les itemsets essentiels (Casali et al., 2005). La fréquence disjonctive d'un itemset X est alors $|\mathcal{O}| - |\overline{cov_{\mathcal{I}}}(X)|$.

L'extraction des motifs minimaux a de nombreuses applications et n'est pas seulement utilisée pour accélérer l'obtention des motifs fréquents. Leurs propriétés sont utiles pour certaines tâches. Par exemple, les motifs minimaux sont utilisés en conjonction de motifs fermés pour produire des règles non-redondantes ou informatives (Zaki, 2000; Pasquier et al., 1999). Les règles séquentielles bénéficient également de la minimalité (Lo et al., 2009). Il est aussi possible d'exploiter les motifs minimaux pour extraire des règles de classification qui sont les éléments clés des classifieurs associatifs (Liu et al., 1998). Notre cadre est bien adapté pour extraire de telles règles de classification qui satisfont une mesure d'intérêt impliquant des fréquences. Supposons que l'ensemble d'objets $\mathcal{O}$ se répartisse en deux classes disjointes $\mathcal{O} = \mathcal{O}_1 \cup \mathcal{O}_2$, la confiance de la règle de classification $X \rightarrow class_1$ est $|\mathcal{O}_1 \cap cov_{\mathcal{I}}(X)|/|cov_{\mathcal{I}}(X)|$. Plus généralement, il est facile de montrer que n'importe quelle mesure fondée sur des fréquences (e.g., le lift, le bond) peut être dérivée des couvertures positive et négative. En plus, les motifs essentiels sont utiles pour dériver les traverses minimales qui correspondent exactement aux motifs maximaux de $\mathcal{M}(\langle (2^{\mathcal{I}}, \mathcal{I}), 2^{\mathcal{I}}, \overline{cov_{\mathcal{I}}}, Id \rangle)$. Rappelons que la génération des traverses minimales est un problème important aux nombreuses applications en logique, en intelligence artificielle et apprentissage (Eiter et Gottlob, 2002; Murakami et Uno, 2013).

Les représentations condensées de motifs minimaux ne sont pas limitées aux seules mesures impliquant la fréquence et au seul langage des itemsets. En effet, il est aussi possible d'extraire des motifs minimaux adéquats à des fonctions d'agrégat comme min,

max ou sum (Soulet et Crémilleux, 2008). Les systèmes minimisables d'ensembles sont également bien adaptés à de telles mesures. Par exemple, considérons la fonction $cov_{min}(X) = \{val(i) | \exists i \in \mathcal{I}, val(i) \leq min(X.val)\}$ qui retourne toutes les valeurs possibles val inférieures à $min(X.val)$. Cette fonction est un opérateur de couverture et $\langle (2^{\mathcal{I}}, \mathcal{I}), 2^{\mathcal{I}}, cov_{\min}, Id \rangle$ est même un système minimisable d'ensembles. Les motifs minimaux adéquats à min correspondent aux motifs minimaux du précédent système. Par ailleurs, la valeur $min(X.val)$ peut être retouvée de cette manière : $\max(cov_{min}(X))$. Une approche similaire fonctionne pour max et sum. En parallèle, plusieurs études ont étendu la notion de générateurs pour prendre en compte d'autres langages comme les séquences (Lo et al., 2008; Gao et al., 2008), les graphes (Zeng et al., 2009). Malheureusement aucun travail ne propose un cadre général pour étendre la minimalité à une large famille de langages comme cela a été fait pour les motifs fermés (Arimura et Uno, 2009). Dans cet article, nous avons fait la connexion entre les systèmes d'ensembles et seulement deux langages : itemsets et chaînes par manque de place. De nombreux autres langages peuvent être représentés en utilisant le formalisme des systèmes d'ensembles. En particulier, tous les langages décrits par Arimura et Uno (2009) conviennent. Bien sûr, les opérateurs de fermetures décrits auparavant pour les itemsets peuvent être étendus à tous ces langages.

6 Conclusion

En proposant la notion nouvelle de système minimisable d'ensembles, ce papier étend le paradigme des motifs minimaux à un large spectre de fonctions et de langages. Ce cadre englobe les principales méthodes comme les motifs libres et essentiels. De plus, DEFME extrait tous les motifs minimaux même dans des bases de données difficiles où les algorithmes de l'état de l'art échouent faute de mémoire suffisante. Des expérimentations ont aussi montré que le concept de minimalité s'avère intéressant pour représenter de manière concise la totalité des motifs en s'appuyant sur un nouveau type de motif, la chaîne minimale. Bien sûr, nous pensons que notre implémentation peut être améliorée même s'il est difficile de trouver un compromis entre méthode générique et vitesse. Nous voulons particulièrement tester la capacité des motifs minimaux à générer des règles de classification pour des types de données encore non explorés. Ensuite, il sera intéressant d'utiliser ces règles minimales afin de construire des classifieurs.

Références

Arimura, H. et T. Uno (2009). Polynomial-delay and polynomial-space algorithms for mining closed sequences, graphs, and pictures in accessible set systems. In *SDM*, pp. 1087–1098.

Boulicaut, J.-F., A. Bykowski, et C. Rigotti (2000). Approximation of frequency queries by means of free-sets. In *PKDD*, Volume 1910 of *LNCS*, pp. 75–85. Springer.

Calders, T. et B. Goethals (2005). Depth-first non-derivable itemset mining. In *SDM*, pp. 250–261.

Calders, T., C. Rigotti, et J.-F. Boulicaut (2004). A survey on condensed representations for frequent sets. In *Constraint-Based Mining and Inductive Databases*, Volume 3848 of *LNCS*, pp. 64–80. Springer.

Casali, A., R. Cicchetti, et L. Lakhal (2005). Essential patterns : A perfect cover of frequent patterns. In *DaWaK*, Volume 3589 of *LNCS*, pp. 428–437. Springer.

Eiter, T. et G. Gottlob (2002). Hypergraph transversal computation and related problems in logic and ai. In *JELIA*, Volume 2424 of *LNCS*, pp. 549–564. Springer.

Gao, C., J. Wang, Y. He, et L. Zhou (2008). Efficient mining of frequent sequence generators. In *WWW*, pp. 1051–1052. ACM.

Hamrouni, T. (2012). Key roles of closed sets and minimal generators in concise representations of frequent patterns. *Intell. Data Anal. 16*(4), 581–631.

Li, J., H. Li, L. Wong, J. Pei, et G. Dong (2006). Minimum description length principle : Generators are preferable to closed patterns. In *AAAI*, pp. 409–414. AAAI Press.

Liu, B., W. Hsu, et Y. Ma (1998). Integrating classification and association rule mining. In *KDD*, pp. 80–86.

Liu, G., J. Li, et L. Wong (2008). A new concise representation of frequent itemsets using generators and a positive border. *Knowl. Inf. Syst. 17*(1), 35–56.

Lo, D., S.-C. Khoo, et J. Li (2008). Mining and ranking generators of sequential patterns. In *SDM*, pp. 553–564. SIAM.

Lo, D., S.-C. Khoo, et L. Wong (2009). Non-redundant sequential rules - theory and algorithm. *Inf. Syst. 34*(4-5), 438–453.

Murakami, K. et T. Uno (2013). Efficient algorithms for dualizing large-scale hypergraphs. In *ALENEX*, pp. 1–13.

Pasquier, N., Y. Bastide, R. Taouil, et L. Lakhal (1999). Efficient mining of association rules using closed itemset lattices. *Inf. Syst. 24*(1), 25–46.

Soulet, A. et B. Crémilleux (2008). Adequate condensed representations of patterns. *Data Min. Knowl. Discov. 17*(1), 94–110.

Szathmary, L., P. Valtchev, A. Napoli, et R. Godin (2009). Efficient vertical mining of frequent closures and generators. In *IDA*, Volume 5772 of *LNCS*, pp. 393–404. Springer.

Zaki, M. J. (2000). Generating non-redundant association rules. In *KDD*, pp. 34–43.

Zeng, Z., J. Wang, J. Zhang, et L. Zhou (2009). FOGGER : an algorithm for graph generator discovery. In *EDBT*, pp. 517–528.

Summary

While the maximal patterns of the equivalence classes have received much attention, the minimal patterns remained in the shadows in particular because of their difficult extraction. In this paper, we present a generic framework for minimal patterns mining by introducing the concept of minimizable set system. This framework addresses various languages such as itemsets or strings, and at the same time, different metrics such as frequency. Then, for any minimizable set system, we introduce a fast minimality checking that is easy to incorporate in a depth-first search algorithm for mining the minimal patterns. We demonstrate that it is polynomial-delay and polynomial-space. Experiments on traditional benchmarks complete our study.