Allier CSPs et motifs locaux pour la découverte de motifs sous contraintes n-aires

Mehdi Khiari, Patrice Boizumault, Bruno Crémilleux

GREYC, CNRS - UMR 6072, Université de Caen,
Campus Côte de Nacre,
F-14032 Caen Cedex, France
{Prenom.Nom}@info.unicaen.fr

Résumé. Dans cet article, nous étudions la relation entre la découverte de motifs sous contraintes et les CSPs (Constraint Satisfaction Problems) afin de définir des contraintes de plus haut niveau qui sont précieuses pour mener à bien des tâches de fouille de données. Pour cela, nous proposons une approche de modélisation et d'extraction de motifs sous contraintes n-aires exploitant les motifs locaux. L'utilisateur définit un ensemble de contraintes n-aires et un solveur de CSP génère l'ensemble des solutions. Notre approche profite des progrès récents sur l'extraction de motifs locaux et permet de modéliser de manière concise et élégante tout ensemble de contraintes combinant plusieurs motifs locaux, permettant ainsi la découverte de motifs répondant mieux aux buts finaux de l'utilisateur. Les expériences menées montrent la faisabilité de notre approche.

1 Introduction

Un problème majeur dans les processus d'Extraction de Connaissances dans les Bases de Données (ECBD) est le nombre conséquent de motifs produits rendant leur utilisation difficile. Ainsi, les motifs les plus significatifs sont souvent noyés au milieu d'informations triviales ou redondantes. D'autre part, l'intérêt d'un motif dépend souvent d'autres motifs et beaucoup de modèles comme les classifieurs ou les méthodes de clustering requièrent d'exploiter et de considérer simultanément plusieurs motifs. Contribuer à la découverte de motifs de plus haut niveau, fondés sur la combinaison de motifs tels que les motifs locaux (cf. section 2.1), permet ainsi la découverte de motifs plus proches des buts finaux de l'utilisateur.

Il existe plusieurs approches visant à réduire les motifs produits, telles que le paradigme de l'extraction sous contraintes Ng et al. (1998), l'utilisation des représentations condensées Calders et al. (2005) ou encore la compression des jeux de données en exploitant le principe du MDL Siebes et al. (2006). En ciblant la recherche de l'information à extraire suivant les centres d'interêt de l'utilisateur, le paradigme des motifs contraints a pour but d'améliorer la qualité des motifs extraits Ng et al. (1998). Ce problème est plutôt bien maîtrisé via des approches génériques de découverte de *motifs locaux* sous contraintes De Raedt et al. (2002); Soulet et Crémilleux (2005). On appelle *contraintes locales* les contraintes visant à extraire les motifs locaux. Le caractère local de la contrainte provient du fait que, vérifier si un motif satisfait ou pas une contrainte donnée, ne dépend pas des autres motifs. Dans la pratique, même si le

nombre de motifs produits est réduit grâce aux contraintes, ce nombre demeure trop important pour une analyse par l'utilisateur. D'autre part, les contraintes locales s'avèrent insuffisantes pour la découverte de motifs de plus haut niveau, tels que des motifs combinant plusieurs motifs locaux. Dans la suite de cet article nous appelons *contraintes n-aires* les contraintes qui portent simultanément sur plusieurs motifs. Ces contraintes permettent d'obtenir des motifs de plus grand intérêt pour l'utilisateur et la construction de modèles globaux issus des données.

L'extraction de motifs sous contraintes locales demande l'exploration d'un espace de recherche de très grande taille, même lorsqu'il s'agit de motifs simples tels que des itemsets. Bien évidemment, l'extraction sous contraintes n-aires est encore plus complexe puisque nous devons prendre en considération et comparer plusieurs motifs à la fois. Certains problèmes particuliers portant sur des contraintes n-aires sont déjà traités Suzuki (2002); Lakshmanan et al. (1998) à l'aide d'algorithmes dédiés à chaque contrainte, mais aucune méthode générique n'est proposée. En parallèle, les problèmes de satisfaction de contraintes (CSPs) permettent de modéliser de façon naturelle des contraintes portant sur plusieurs variables Apt et Wallace (2007).

Cet article propose une approche combinant la fouille de motifs locaux et la programmation par contraintes (PPC) pour la résolution de contraintes n-aires. Plus généralement, il explore les relations entre l'extraction de motifs contraints et la PPC. Nous montrons qu'on peut modéliser un problème d'extraction de motifs sous contraintes n-aires à l'aide d'un CSP en assimilant chaque motif recherché à une variable. Une requête d'un utilisateur correspond alors à un ensemble de contraintes n-aires décrivant les motifs recherchés. Le grand avantage de cette modélisation est sa flexibilité qui permet de traiter un large ensemble de contraintes et donc de motifs. D'autre part, nous souhaitons aussi profiter des avancées récentes dans le domaine de l'extraction de motifs locaux et de la puissance des extracteurs pour ce type de motifs. Ainsi, notre approche partitionne une requête de l'utilisateur en deux ensembles : le premier rassemble les contraintes locales qu'il est possible d'inférer de la requête, et qui sont alors résolues par un extracteur de motifs locaux avant et indépendamment du deuxième ensemble, ce dernier contenant les contraintes n-aires qui seront résolues par un solveur de CSP. C'est cette combinaison entre les niveaux local et n-aire qui nous permet la découverte de motifs sous contraintes n-aires.

Cet article est organisé de la manière suivante. La section 2 présente le contexte et les motivations de ce travail. Un état de l'art en extraction de motifs et PPC est présenté en section 3. La section 4 décrit notre approche pour extraire des motifs sous contraintes n-aires. La section 5 présente les résultats expérimentaux. La section 6 conclut et propose des perspectives.

2 Définitions et motivations

2.1 Definitions

Soit $\mathcal{I}$ un ensemble de littéraux distincts appelés *items*, un motif ensembliste[1] d'items correspond à un sous-ensemble non vide de $\mathcal{I}$. Ces motifs sont regroupés dans le langage $\mathcal{L}_\mathcal{I} = 2^\mathcal{I} \backslash \emptyset$. Un contexte transactionnel est alors défini comme un multi-ensemble de motifs de $\mathcal{L}_\mathcal{I}$. Chacun de ces motifs, appelé transaction, constitue une entrée de la base de données. Ainsi, le tableau 1 présente un contexte transactionnel $\boldsymbol{r}$ où 9 transactions étiquetées $o_1, \ldots, o_9$ sont décrites par 6 items $A, \ldots, D, c_1, c_2$.

[1] Dans cet article, nous nous intéressons au cas des motifs ensemblistes

Trans.	Items
o_1	$A\ B\quad\quad c_1$
o_2	$A\ B\quad\quad c_1$
o_3	$\quad\ C\quad c_1$
o_4	$\quad\ C\quad c_1$
o_5	$\quad\ C\quad c_1$
o_6	$A\ B\ C\ D\quad c_2$
o_7	$\quad\ C\ D\quad c_2$
o_8	$\quad\ C\quad\quad c_2$
o_9	$\quad\quad\ D\quad c_2$

TAB. 1 – *Exemple de contexte transactionnel* ***r****.*

L'extraction sous contraintes cherche la collection de tous les motifs X de $\mathcal{L}_{\mathcal{I}}$ présents dans ***r*** et satisfaisant un prédicat appelé *contrainte*. Ces motifs sont appelés *motifs locaux*, ce sont des régularités observées dans certaines parties des données. Un motif local est particulièrement intéressant s'il présente un comportement déviant par rapport au modèle global des données Hand (2002). Plusieurs contraintes sont utilisées pour évaluer la pertinence des motifs locaux. La contrainte la plus classique est certainement celle de fréquence minimale qui permet d'extraire des motifs dont la fréquence dans le jeu de données est plus grande qu'un seuil $\gamma > 0$ donné : $freq(X) \geq \gamma$. Plusieurs travaux Ng et al. (1998) utilisent d'autres mesures d'intérêt telle que par exemple la mesure d'aire ($aire(X)$ est égale à la fréquence de X multipliée par sa longueur, i.e., $aire(X) = freq(X) \times count(X)$ où $count(X)$ représente la cardinalité de X).

En pratique, l'utilisateur est souvent intéressé par la découverte de motifs de plus haut niveau, comme par exemple les règles de classification les plus simples afin d'éviter le surapprentissage Yin et Han (2003), ou encore des paires de règles d'exception Suzuki (2002) capables de révéler des caractéristiques globales des données. De tels motifs reposent sur des propriétés impliquant plusieurs motifs locaux et sont formalisés par la notion de *contrainte n-aire*, notion qui dépasse clairement celle de contrainte locale :

Définition 2.1 (Contrainte n-aire) *Une contrainte q est dite n-aire si sa vérification nécessite de comparer plusieurs motifs entre eux.*

2.2 Contexte et motivations

Cette section montre l'intérêt des contraintes n-aires en donnant des exemples de motifs s'exprimant à l'aide de contraintes n-aires. Ces motifs sont plus significatifs que de simples motifs locaux et sont ainsi plus recherchés par l'utilisateur. Par exemple, E. Suzuki s'est intéressé à la découverte de paires de règles incluant une règle d'exception Suzuki (2002) (aussi appelé règles d'exception). Une règle d'exception est définie comme étant une règle qui dévie d'un comportement général exprimé par une autre règle : l'intérêt de cette définition est d'expliciter la caractéristique d'exception par rapport à un comportement général. De façon plus formelle, les règles d'exception sont définies comme suit (I est un item, par exemple une valeur de classe, X et Y sont des motifs locaux) :

$$exception(X \rightarrow \neg I) \equiv \begin{cases} vrai & \text{si } \exists Y \in \mathcal{L}_{\mathcal{I}} \text{ tel que } Y \subset X, \text{ on a } (X \backslash Y \rightarrow I) \wedge (X \rightarrow \neg I) \\ faux & \text{sinon} \end{cases}$$

Dans une telle paire de règles, $X \rightarrow \neg I$ est une règle d'exception, puisque on a généralement $X \backslash Y \rightarrow I$. La règle d'exception révèle ainsi une information inattendue. Cette définition suppose que la règle générale est de forte fréquence et de forte confiance tandis que la règle d'exception est rare mais de très forte confiance (la confiance d'une règle $X \rightarrow Y$ est $freq(X \cup Y)/freq(X)$). La comparaison entre règles d'une paire signifie qu'une telle paire ne peut pas être modélisée par une approche uniquement à base de motifs locaux, alors que nous verrons qu'elle se modélise aisément avec des contraintes n-aires. Donnons un exemple de règle d'exception à partir du tableau 1 : en prenant 2/3 comme seuil de confiance d'une règle, la règle $AC \rightarrow \neg c_1$ est une règle d'exception puisque nous avons conjointement $A \rightarrow c_1$ et $AC \rightarrow \neg c_1$. Suzuki a proposé une méthode fondée sur une estimation probabiliste Suzuki (2002) pour extraire de telles paires de règles, mais elle demeure dédiée à ce type de motifs.

Considérons maintenant le domaine de l'analyse d'expression de gènes pour donner un autre exemple de motifs définis par des contraintes n-aires. Dans ce contexte, les motifs locaux composés de tags (ou gènes) et satisfaisants la contrainte d'aire (cf. section 2.1) sont au cœur de la recherche de groupes de synexpression Kléma et al. (2008). Néanmoins, dans des données réelles telles que celles du transcriptome, la recherche de motifs tolérants aux erreurs y est capitale afin de tenir compte de l'incertain qui est présent dans les données Besson et al. (2006). Les contraintes n-aires sont une façon naturelle de concevoir des motifs tolérants aux erreurs candidats à être des groupes de synexpression : ceux-ci sont définis par l'union de plusieurs motifs locaux satisfaisant la contrainte d'aire et ayant un fort recouvrement entre eux. Plus précisément, à partir de deux motifs locaux X et Y, on définit la contrainte n-aire suivante $aire(X) > min_{aire} \wedge aire(Y) > min_{aire} \wedge (aire(X \cap Y) > \alpha \times min_{aire})$ où min_{aire} est le seuil d'aire minimale et α est un paramètre donné par l'utilisateur pour fixer le recouvrement minimal entre motifs locaux. Cette contrainte n-aire peut être étendue à un nombre indéfini de motifs par l'utilisation du quantificateur universel (cf. section 6). La section 4 présente notre approche pour modéliser les motifs satisfaisant de telles contraintes et comment on peut les extraire combinant l'extraction de motifs locaux et PPC.

3 État de l'art et programmation par contraintes

3.1 Motifs locaux et découverte de motifs en fouille de données

Plusieurs travaux traitent de la découverte de motifs locaux sous contraintes. Leur idée centrale est d'exploiter la propriété de monotonie entre motifs qui permet de puissants élagages dans les espaces de recherche Mannila et Toivonen (1997). Nous utilisons ici le prototype MUSIC-DFS[2] parce qu'il permet d'extraire efficacement l'ensemble correct et complet des motifs selon un large éventail de contraintes locales, telle que par exemple la contrainte d'aire. Les motifs locaux sont produits sous forme de représentation condensée composée d'intervalles disjoints où chaque intervalle synthétise un ensemble de motifs satisfaisants la contrainte Soulet et al. (2006).

Il existe quelques approches qui considèrent simultanément plusieurs motifs locaux : les "patterns teams" Knobbe et Ho (2006), les ensembles sous contraintes de motifs De Raedt et Zimmermann (2007) ou encore la sélection de motifs selon leur apport compte tenu des autres

[2] http://www.info.univ-tours.fr/~soulet/music-dfs/music-dfs.html

motifs déjà sélectionnés Bringmann et Zimmermann (2007). Même si ces approches comparent explicitement les motifs entre eux, elles sont principalement fondées sur la réduction de la redondance entre motifs ou poursuivent des objectifs spécifiques tels que la classification. De part leur flexibilité, les contraintes n-aires permettent à l'utilisateur, avec un même cadre de modélisation, d'exprimer des biais de recherche variés et donc des types de motifs très divers. Notons qu'il existe des cadres génériques pour la construction de modèles globaux à partir de motifs locaux Knobbe et al. (2008); Giacometti et al. (2009), mais ces cadres ne proposent pas de méthode, ils permettent de mieux comparer celles existantes. De façon plus générale, nous pensons que notre approche illustre l'intérêt de la PPC dans cette problématique générale de construction de modèles globaux à partir de motifs locaux.

La PPC est un puissant paradigme déclaratif pour la modélisation et la résolution de problèmes combinatoires. Une première approche utilisant la PPC pour l'extraction de motifs locaux a été proposée dans De Raedt et al. (2008) pour modéliser des contraintes telles que la fréquence, la fermeture, ou d'autres contraintes monotones et anti-monotones ou des combinaisons de telles contraintes. Les motifs satisfaisants de telles contraintes sont obtenus avec le solveur de contraintes Gecode[3] en utilisant la programmation linéaire en 0/1. Mais cette approche ne traite pas le cas des motifs provenant de relations entre plusieurs motifs locaux tels que ceux décrits dans la section 2.

3.2 CSP Ensembliste

Un CSP est un triplet $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ où $\mathcal{X}$ est un ensemble fini de variables, $\mathcal{D}$ est un ensemble de domaines finis et $\mathcal{C}$ est l'ensemble des contraintes qui restreignent les valeurs que peuvent prendre simultanément les variables. Il existe plusieurs types de CSPs : numériques, booléens, ensemblistes, etc. Ils diffèrent principalement par la nature de leurs domaines et leurs algorithmes de filtrage. Dans cette section nous présentons les CSPs ensemblistes (CSPEs) qui seront utilisés dans notre modélisation ainsi qu'un bref aperçu des règles de filtrage associées.

Définition 3.1 (Intervalle ensembliste) *Soient lb et ub deux ensembles tels que $lb \subset ub$, l'intervalle ensembliste $[lb..ub]$ est défini comme suit : $[lb..ub] = \{E \text{ tel que } lb \subseteq E \text{ et } E \subseteq ub\}$.*

Les intervalles ensemblistes permettent de remédier aux problèmes de stockage de données en condensant la représentation des valeurs possibles des variables. Par exemple : $[\{1\}..\{1,2,3\}]$ est égal à $\{\{1\},\{1,2\},\{1,3\},\{1,2,3\}\}$ et $[\{\}..\{1,2,3\}]$ est égal à $2^{\{1,2,3\}}$.

Définition 3.2 (CSP ensembliste) *Un CSPE est un triplet $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ où $\mathcal{C} = \{c_1, ..., c_m\}$ est un ensemble de contraintes associées à un ensemble de variables $\mathcal{X} = \{X_1, ..., X_n\}$. Chaque variable X_i admet un domaine initial D_{X_i} de valeurs sous forme d'intervalle ensembliste (ou union d'intervalles ensemblistes) et $D = \{D_{X_1}, ..., D_{X_n}\}$.*

Règles de filtrage pour les CSPE : Pour les CSPs, le filtrage consiste à réduire les domaines des variables en supprimant des valeurs qui ne peuvent appartenir à une solution. Dès qu'un domaine D_{X_i} est vide (il n'y a pas de valeur viable pour X_i), un échec est généré pour la recherche. Les règles de filtrage pour les CSPs numériques et les CSPEs sont présentés dans Lhomme (1993); Gervet (1997).

[3] `http://www.gecode.org`

A titre d'exemple, la règle de filtrage pour les intervalles ensemblistes pour la contrainte d'*intersection* est la suivante : Soient $D_x = [a_x..b_x]$, $D_y = [a_y \,..\, b_y]$ et $D_z = [a_z \,..\, b_z]$ trois domaines représentés par des intervalles ensemblistes, leurs domaines filtrés par rapprort à la contrainte $Z = X \cap Y$ sont respectivement $D'_x = [a_x \cup a_z \,..\, b_x \setminus ((b_x \cap a_y) \setminus b_z]$, $D'_y = [a_y \cup a_z \,..\, b_y \setminus ((b_y \cap a_x) \setminus b_z]$ et $D'_z = [a_z \cup (a_x \cap a_y) \,..\, b_z \cap b_x \cap b_y]$ si $((b_x \cap b_y) \subset b_z \wedge (b_x \cap b_y) \neq \emptyset)$ et $D'_x = D'_y = D'_z = \emptyset$ sinon.

Outil de programmation : ECL^iPS^e [4] est un outil de PPC supportant les techniques les plus connues pour la résolution des problèmes de satisfaction et d'optimisation de contraintes. ECL^iPS^e est basé sur le paradigme de la programmation logique par contraintes Apt et Wallace (2007). Différents types de contraintes tels que les contraintes ensemblistes et les contraintes numériques peuvent être utilisées simultanément. Pour la résolution de CSPs ensemblistes, ECL^iPS^e propose différentes librairies tels que *ic-sets* et *conjunto* Gervet (1997).

4 La PPC pour la découverte de motifs

Notre approche se base sur trois points clés : (i)les larges possibilités de modélisation et de résolution offertes par les CSPs, en particulier les CSPEs et les CSPs numériques, (ii)les récents progrès dans le domaine de la fouille de motifs locaux, et (iii)le fait que les contraintes locales peuvent être résolues avant et indépendamment des contraintes n-aires. Dans cette section, nous donnons une vue d'ensemble de notre approche avant de détailler chacune de ces trois étapes en considérant l'exemple des règles d'exception (section 2.2).

4.1 Vue d'ensemble de notre approche

La figure 1 présente une vue d'ensemble des trois étapes de notre approche :

1. Modéliser la requête sous forme de CSPs puis diviser les contraintes en locales et n-aires.
2. Résoudre les contraintes locales à l'aide d'un extracteur de motifs locaux (MUSIC-DFS, cf. section 3.1) qui produit une représentation condensée par intervalles de tous les motifs satisfaisant les contraintes locales.
3. Résoudre les contraintes n-aires à l'aide d'un solveur de contraintes (ECL^iPS^e, cf. section 3.2). Le domaine de chaque variable résulte de la représentation condensée par intervalles (calculée dans Etape-2).

4.2 Etape-1 : Modélisation de la requête sous forme de CSPs

Soit ***r*** un jeu de données ayant nb transactions et $\mathcal{I}$ l'ensemble de ses items. On modélise le problème à l'aide de deux CSPs $\mathcal{P}$ et $\mathcal{P}'$ inter-reliés :

1. CSP ensembliste $\mathcal{P} = (\mathcal{X}, \mathcal{D}, \mathcal{C})$ où :
   - $\mathcal{X} = \{X_1, ..., X_n\}$. Chaque variable X_i représente un motif inconnu.
   - $\mathcal{D} = \{D_{X_1}, ..., D_{X_n}\}$. Le domaine initial de chaque variable X_i est $[\{\} \,..\, \mathcal{I}]$.
   - $\mathcal{C}$ est une conjonction de contraintes ensemblistes formulées à l'aide d'opérateurs ensemblistes ($\cup$, $\cap$, $\setminus$, $\in$, $\notin$, ...).

[4] http://www.eclipse-clp.org

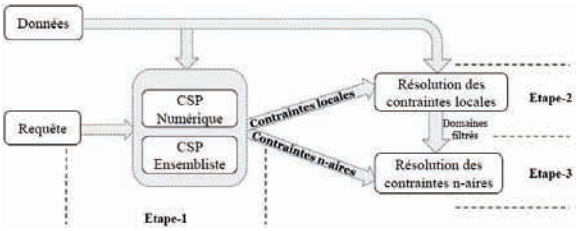

FIG. 1 – *Vue d'ensemble des 3 étapes de la résolution*

2. CSP numérique $\mathcal{P}' = (\mathcal{F}, \mathcal{D}', \mathcal{C}')$ où :
   - $\mathcal{F} = \{F_1, ..., F_n\}$. Chaque variable F_i est la fréquence du motif X_i.
   - $\mathcal{D}' = \{D_{F_1}, ..., D_{F_n}\}$. Le domaine initial de chaque variable F_i est $[1\ ..\ nb]$.
   - $\mathcal{C}'$ est une conjonction de contraintes numériques formulées à l'aide d'opérateurs numériques ($<,\ \leq,\ =, +, ...$).

L'ensemble de toutes les contraintes ($\mathcal{C} \cup \mathcal{C}'$) est divisé en deux sous ensembles :
- $\mathcal{C}_{loc}$ est l'ensemble des contraintes locales à résoudre (par MUSIC-DFS). Les solutions seront sous la forme de représentation condensée par intervalles.
- $\mathcal{C}_{n-aire}$ est l'ensemble des contraintes n-aires à résoudre (par ECL^iPS^e), où les domaines des variables X_i et F_i sont déduits de la représentation condensée par intervalles calculée dans l'étape précédente.

Les contraintes locales sont résolues avant et indépendamment des contraintes n-aires. L'espace de recherche pour les contraintes n-aires est ainsi réduit à l'espace des solutions issu des contraintes locales.

4.3 Exemple : modélisation des règles d'exception

4.3.1 Reformulation :

Soit $freq(X)$ la valeur de la fréquence du motif X. Soient I et $\neg I \in \mathcal{I}$ deux items (dans cet exemple, I et $\neg I$ sont deux classes du jeu de données). Soient $\gamma_1, \gamma_2, \delta_1, \delta_2 \in \mathbb{N}$. La contrainte des règles d'exception (section 2.2) peut être formulée comme suit :
- $X \backslash Y \rightarrow I$ peut être exprimée par la conjonction : $freq((X \setminus Y) \sqcup^5 I) \geq \gamma_1\ \wedge$ $(freq(X \setminus Y) - freq((X \setminus Y) \sqcup I)) \leq \delta_1$ qui signifie que $X \backslash Y \rightarrow I$ est une règle fréquente et de forte confiance.
- $X \rightarrow \neg I$ peut être exprimée par la conjonction : $freq(X \sqcup \neg I) \leq \gamma_2\ \wedge (freq(X) - freq(X \sqcup \neg I)) \leq \delta_2$ qui signifie que $X \rightarrow \neg I$ est une règle rare et de forte confiance.

Pour résumer :

$$exception(X \rightarrow \neg I) \equiv \begin{cases} \exists Y \subset X \text{ tel que :} \\ freq((X \setminus Y) \sqcup I) \geq \gamma_1\ \wedge (freq(X \setminus Y) - freq((X \setminus Y) \sqcup I)) \leq \delta_1\ \wedge \\ freq(X \sqcup \neg I) \leq \gamma_2\ \wedge (freq(X) - freq(X \sqcup \neg I)) \leq \delta_2 \end{cases}$$

[5] le symbole $\sqcup$ est l'opérateur d'union disjointe

4.3.2 Modélisation sous forme de CSP :

Les variables des CSPs correspondants sont définies comme suit :

- Variables ensemblistes $\{X_1, X_2, X_3, X_4\}$représentant les motifs recherchés :
  - $X_1 : X \setminus Y$,
  - $X_2 : (X \setminus Y) \sqcup I$ (règle générale),
  - $X_3 : X$,
  - $X_4 : X \sqcup \neg I$ (règle d'exception).

  Variables numériques $\{F_1, F_2, F_3, F_4\}$ représentant les valeurs des fréquences des motifs recherchés (la variable F_i correspond à la fréquence du motif représenté par X_i).

Le tableau 2 décrit l'ensemble des contraintes modélisant les règles d'exception.

Contraintes	Formulation CSP	Locales	N-aires
$freq((X \setminus Y) \sqcup I) \geq \gamma_1$	$F_2 \geq \gamma_1$	$\times$	
	$\wedge \quad I \in X_2$	$\times$	
	$\wedge \quad X_1 \subsetneq X_3$		$\times$
$freq(X \setminus Y) - freq((X \setminus Y) \sqcup I) \leq \delta_1$	$F_1 - F_2 \leq \delta_1$		$\times$
	$\wedge \quad X_2 = X_1 \sqcup I$		$\times$
$freq(X \sqcup \neg I) \leq \gamma_2$	$F_4 \leq \gamma_2$	$\times$	
	$\wedge \quad \neg I \in X_4$	$\times$	
$freq(X) - freq(X \sqcup \neg I) \leq \delta_2$	$F_3 - F_4 \leq \delta_2$		$\times$
	$\wedge \quad X_4 = X_3 \sqcup \neg I$		$\times$

TAB. 2 – *La modélisation des règles d'exception*

4.3.3 Récapitulatif :

- CSPE : $\mathcal{C} = \{(I \in X_2), (X_2 = X_1 \sqcup I), (\neg I \in X_4), (X_4 = X_3 \sqcup \neg I), (X_1 \subsetneq X_3\})$
- CSP numérique : $\mathcal{C}' = \{(F_2 \geq \gamma_1), (F_1 - F_2 \leq \delta_1), (F_4 \leq \gamma_2), (F_3 - F_4 \leq \delta_2)\}$
- $\mathcal{C}_{loc} = \{(I \in X_2), (F_2 \geq \gamma_1), (F_4 \leq \gamma_2), (\neg I \in X_4)\}$
- $\mathcal{C}_{n-aire} = \{(F_1 - F_2 \leq \delta_1), (X_2 = X_1 \sqcup I), (F_3 - F_4 \leq \delta_2), (X_4 = X_3 \sqcup \neg I), (X_1 \subsetneq X_3)\}$

4.4 Etape-2 : Résolution des contraintes locales

Dans le but de bénéficier pleinement de l'efficacité de la fouille de motifs locaux, l'ensemble des contraintes locales $\mathcal{C}_{loc}$ est divisé en une union disjointe de $\mathcal{C}_i$ (pour $i \in [1..n]$) où chaque $\mathcal{C}_i$ est l'ensemble des contraintes portant sur X_i et F_i. Chaque C_i peut être résolu séparément (nous utilisons MUSIC-DFS, cf. section 3.1). Soit CR_i la représentation condensée par intervalles de toutes les solutions de $\mathcal{C}_i$. $CR_i = \bigcup_p (f_p, I_p)$ où I_p est un intervalle ensembliste vérifiant : $\forall x \in I_p,\ freq(x) = f_p$. Ensuite, les domaines réduits (voir section 4.3.2) pour les variables X_i et F_i sont :

- D_{F_i} : l'ensemble de tous les f_p dans CR_i
- D_{X_i} : l'union de tous les I_p dans CR_i

4.5 Etape-3 : Résolution des contraintes n-aires

Les domaines des variables X_i et F_i (pour $i \in \{1, 2, 3, 4\}$) sont obtenus à partir de la représentation condensée des motifs satisfaisant toutes les contraintes locales. La résolution par ECL^iPS^e permet ainsi d'obtenir toutes les solutions satisfaisant l'intégralité des contraintes (locales et n-aires).

5 Expérimentations

L'étude expérimentale porte sur le jeu de données *postoperative-patient-data* de UCI repository[6]. Ce jeu de données comporte 90 transactions décrites par 23 items et est caractérisé par deux classes (deux transactions appartenant à une troisième classe n'ont pas été considérées). Nous testons notre approche avec la contrainte des règles d'exception (dans ce qui suit, l'item I donné dans la définition des règles d'exception représente une des deux classes du jeu de données). Toutes les expériences ont été effectuées sur une machine dotée d'un processeur Intel Centrino Duo 2 GHz et de 2GB de mémoire RAM sous Linux. De façon générale, ces expériences montrent la faisabilité de notre approche.

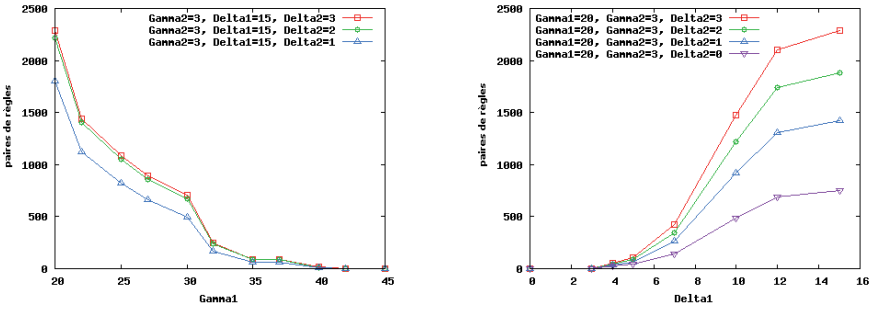

FIG. 2 – *Nombre de paires de règles en fonction de γ_1 (gauche) et δ_1 (droite)*

La figure 2 donne le nombre de paires de règles d'exception en fonction de γ_1 (partie gauche de la figure) et de δ_1 (partie droite de la figure). Plusieurs combinaisons de paramètres ont été testées. Comme attendu, en baissant la valeur de γ_1 on obtient un nombre plus élevé de paires de règles. On constate un comportement similaire en faisant varier δ_1 (partie droite de la figure 2) : le nombre de paires de règles augmente en fonction de δ_1 (quand δ_1 croit, la valeur de la confiance des règles générales décroît, d'où l'obtention d'un plus grand nombre de paires). Il est intéressant de noter que ces courbes mettent aussi en valeur la facilité de contrôle de qualité des règles en modifiant les valeurs des paramètres. Par exemple, pour $(\gamma_1 = 20, \delta_1 = 5, \gamma_2 = 1, \delta_2 = 0)$, on obtient 25 paires où la confiance de la règle générale est supérieure ou égale à 83% et des règles d'exception exactes (confiance égale à 100%) : on constate qu'on obtient ainsi des paires de règles de très bonne qualité. De plus, notre approche permet aisément l'ajout

[6] www.ics.uci.edu/~mlearn/MLRepository.html

de nouvelles propriétés, toujours sous forme de contraintes, comme par exemple le contrôle de la taille des règles générales relativement à celle des règles d'exception (par exemple, au plus 3 items supplémentaires pour la règle d'exception par rapport à la règle générale).

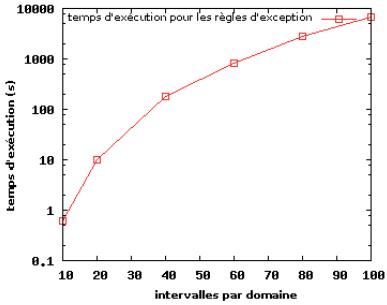

FIG. 3 – *Temps d'exécution en fonction du nombre d'intervalles par domaine*

La figure 3 montre l'évolution du temps d'exécution de notre approche en fonction du nombre d'intervalles par domaine. Pour chaque point de la courbe, les quatre variables ont le même domaine, et par conséquent le même nombre d'intervalles par domaine. Evidemment, le temps d'exécution augmente en fonction du nombre d'intervalles (on utilise une échelle algorithmique pour l'axe Y). Il est intéressant de noter que le temps d'exécution diminue quand on cherche des paires de règles de meilleure qualité. En effet, la recherche de règles générales de haute fréquence et des règles d'exception rares permet un meilleur élagage de l'espace de recherche et par suite un nombre d'intervalles moins important. Le tableau 3 décrit le nombre d'intervalles du domaine de la variable X_2 (voir section 4.3.2) par rapport à différentes contraintes locales. Ceci montre l'intérêt de la résolution séparée des contraintes locales.

Contrainte locale	Nombre d'intervalles dans D_{X_2}
-	3002
$I \in X_2$	1029
$I \in X_2 \wedge freq(X_2) >= 20$	52
$I \in X_2 \wedge freq(X_2) >= 25$	32

TAB. 3 – *Nombre d'intervalles pour différentes contraintes locales (cas de D_{X_2})*

Discussion Dans le but d'établir la consistance aux bornes, les solveurs de CSPE approximent l'union de deux intervalles par leur enveloppe convexe. L'enveloppe convexe de $[lb_1 .. ub_1]$ et $[lb_2 .. ub_2]$ est définie par $[lb_1 \cap lb_2 .. ub_1 \cup ub_2]$. Ainsi, si le filtrage s'applique de nombreuses fois sur le domaine d'une même variable, ce domaine peut rapidement être approximé par l'ensemble de tous les motifs possibles $[\emptyset .. \mathcal{I}]$. *Pour contourner ce problème,* pour chaque variable X_i dont la représentation condensée de motifs satisfaisant les contraintes locales est $CR_i = \bigcup_p (f_p, I_p)$, la recherche est établie successivement dans chaque I_p. Cette approche est correcte et complète. En revanche, on ne profite pas pleinement du filtrage puisque la suppression d'une valeur est propagée uniquement dans les intervalles traités et non dans la totalité

des domaines. Ceci explique les résultats de la section 5 qui montrent que les temps d'exécution augmentent fortement en fonction du nombre d'intervalles. Une première solution serait d'implémenter un opérateur d'union ensembliste plus adapté au cœur du solveur de CSPE. Une autre possibilité serait d'utiliser une représentation condensée non exacte dans le but de réduire le nombre d'intervalles produits.

6 Conclusions et perspectives

Nous avons présenté une approche générale pour la découverte de motifs sous contraintes n-aires. Celle-ci permet de modéliser, de façon flexible, tout ensemble de contraintes combinant plusieurs motifs locaux. L'ensemble correct et complet de toutes les solutions satisfaisant ces contraintes est ainsi obtenu grâce à une résolution en deux phases dont la première profite de l'efficacité des extracteurs de motifs locaux pour la résolution des contraintes locales, tandis que la deuxième bénéficie de la puissance des solveurs de CSPs pour la résolution des contraintes n-aires. La combinaison de ces deux approches permet de formuler aisément des contraintes permettant la découverte de motifs de plus haut niveau et répondant mieux aux buts finaux de l'utilisateur. Les expérimentations montrent la faisabilité de notre approche.

Dans les CSPs classiques, toutes les variables sont quantifiées existentiellement. Or la formulation de contraintes importantes telle que la contrainte de *peak* (un motif est considéré comme *peak* si tous ses voisins ont une valeur, par rapport à une mesure, inférieure à un seuil donné) nécessite l'usage de quantificateurs universels. Nous pensons que les CSPs quantifiées (QCSP) Benhamou et Goualard (2000) sont une voie prometteuse.

Remerciements. Nous remercions Arnaud Soulet pour les discussions fructueuses et MUSIC-DFS. Ce travail est partiellement financé par l'ANR (projet Bingo 2 ANR-07-MDCO-014).

Références

Apt, K. R. et M. Wallace (2007). *Constraint Logic Programming using Eclipse*. New York, NY, USA : Cambridge University Press.

Benhamou, F. et F. Goualard (2000). Universally quantified interval constraints. In *proceedings of CP'00*, London, UK, pp. 67–82. Springer-Verlag.

Besson, J., C. Robardet, et J.-F. Boulicaut (2006). Mining a new fault-tolerant pattern type as an alternative to formal concept discovery. In *ICCS'06*, Aalborg, Denmark, pp. 144–157.

Bringmann, B. et A. Zimmermann (2007). The chosen few : On identifying valuable patterns. In *proceedings of ICDM-07*, Omaha, NE, pp. 63–72.

Calders, T., C. Rigotti, et J.-F. Boulicaut (2005). A survey on condensed representations for frequent sets. In *Constraint-Based Mining and Inductive Databases*, Volume 3848 of *LNAI*.

De Raedt, L., T. Guns, et S. Nijssen (2008). Constraint Programming for Itemset Mining. In *KDD'08*, Las Vegas, Nevada, USA.

De Raedt, L., M. Jäger, S. D. Lee, et H. Mannila (2002). A theory of inductive query answering. In *proceedings of ICDM'02*, Maebashi, Japan, pp. 123–130.

De Raedt, L. et A. Zimmermann (2007). Constraint-based pattern set mining. In *Proceedings of the Seventh SIAM International Conference on Data Mining*, Minneapolis, USA.

Gervet, C. (1997). Interval Propagation to Reason about Sets : Definition and Implementation of a Practical Language. *Constraints 1*(3), 191–244.

Giacometti, A., E. Khanjari Miyaneh, P. Marcel, et A. Soulet (2009). A framework for pattern-based global models. In *10th IDEAL Int. Conf.*, Burgos, Spain, pp. 433–440.

Hand, D. J. (2002). *ESF exploratory workshop on Pattern Detection and Discovery in Data Mining*, Volume 2447 of *LNCS*, Chapter Pattern detection and discovery, pp. 1–12. Springer.

Kléma, J., S. Blachon, A. Soulet, B. Crémilleux, et O. Gandrillon (2008). Constraint-based knowledge discovery from sage data. *In Silico Biology 8(0014)*.

Knobbe, A., B. Crémilleux, J. Fürnkranz, et M. Scholz (2008). From local patterns to global models : The lego approach to data mining. In *International Workshop "From Local Patterns to Global Models" co-located with ECML/PKDD'08*, Antwerp, Belgium, pp. 1–16.

Knobbe, A. et E. Ho (2006). Pattern teams. In J. Fürnkranz, T. Scheffer, et M. Spiliopoulou (Eds.), *proceedings of PKDD'06*, Berlin, Germany, pp. 577–584. Springer-Verlag.

Lakshmanan, L. V., R. Ng, J. Hah, et A. Pang (1998). Optimization of constrained frequent set queries with 2-variable constraints.

Lhomme, O. (1993). Consistency Techniques for Numeric CSPs. In *Proc. of the 13th IJCAI*, Chambery, France, pp. 232–238.

Mannila, H. et H. Toivonen (1997). Levelwise search and borders of theories in knowledge discovery. *Data Mining and Knowledge Discovery 1*, 241–258.

Ng, R. T., V. S. Lakshmanan, J. Han, et A. Pang (1998). Exploratory mining and pruning optimizations of constrained associations rules. In *proceedings of ACM SIGMOD'98*.

Siebes, A., J. Vreeken, et M. van Leeuwen (2006). Item sets that compress. In *Proceedings of the Sixth SIAM International Conference on Data Mining*, Bethesda, MD, USA. SIAM.

Soulet, A. et B. Crémilleux (2005). An efficient framework for mining flexible constraints. In *Proceedings of PAKDD'05*, Volume 3518 of *LNAI*, Hanoi, Vietnam, pp. 661–671. Springer.

Soulet, A., J. Kléma, et B. Crémilleux (2006). Efficient mining under rich constraints derived from various datasets. In *KDID*, Volume 4747 of *LNCS*, pp. 223–239. Springer.

Suzuki, E. (2002). Undirected Discovery of Interesting Exception Rules. *International Journal of Pattern Recognition and Artificial Intelligence 16*, 1065–1086.

Yin, X. et J. Han (2003). CPAR : classification based on predictive association rules. In *proceedings of SDM'03*, San Fransisco, CA.

Summary

In this paper, we investigate the relationship between local constraint-based mining and constraint satisfaction problems and we propose an approach to model and mine patterns combining several local patterns, i.e., patterns defined by n-ary constraints. The user specifies a set of n-ary constraints and a constraint solver generates the whole set of solutions. Our approach takes benefit from the recent progress on mining local patterns by pushing with a solver on local patterns all local constraints which can be inferred from the n-ary ones. Our approach enables to model in a flexible way *any* set of constraints combining several local patterns. Experiments show the feasibility of our approach.