Une approche PPC pour la fouille de données séquentielles

Jean-Philippe Métivier*, Samir Loudni*, Thierry Charnois**

*GREYC (CNRS UMR 6072) – Université de Caen
Campus II Côte de Nacre, 14000 Caen - France
** LIPN (CNRS UMR 7030) – Université PARIS 13
99, avenue Jean-Baptiste Clément 93430 Villetaneuse - France

Résumé. Nous proposons dans cet article une nouvelle approche croisant des techniques de programmation par contraintes et de fouille pour l'extraction de motifs séquentiels. Le modèle que nous proposons offre un cadre générique et déclaratif pour modéliser et résoudre des contraintes de nature hétérogène.

1 Introduction

Introduite par (Agrawal et Srikant, 1995), la fouille de données séquentielles permet de découvrir des corrélations entre des événements selon une relation d'ordre (e.g. le temps). En intégrant des connaissances sous forme d'a priori dans le processus de fouille, l'extraction de motifs sous contraintes contribue à réduire le nombre de motifs en ciblant les motifs potentiellement intéressants (Dong et Pei, 2007). De plus, elle permet souvent de concevoir des algorithmes plus efficaces en réduisant l'espace de recherche. De nombreux algorithmes sont proposés dans la littérature pour l'extraction de motifs séquentiels (Dong et Pei, 2007). Malheureusement, ces méthodes ne traitent que quelques classes particulières de contraintes (monotonicité et anti-monotonicité) avec des techniques dédiées. Ce manque de généricité est un frein à la découverte de motifs pertinents car chaque nouveau type de contraintes entraîne la conception et le développement d'une méthode ad hoc.

Pour lever ce frein, des travaux récents visent à croiser les techniques de Programmation Par Contraintes (PPC) et de fouille pour l'extraction sous contraintes de motifs d'itemsets (Guns et al., 2011; Khiari et al., 2010). Le point commun de ces travaux est de modéliser le problème de la fouille de motifs en un problème de CSP. Une telle modélisation présente l'avantage d'être flexible en permettant de définir de nouvelles contraintes sans s'occuper de leur résolution. Mais, les méthodes proposées sont conçues pour des données ensemblistes et la dimension séquentielle reste quasiment non exploitée, à l'exception des travaux de (Coquery et al., 2012) qui portent sur un cas particulier de chaîne (et non sur une base de séquences).

L'originalité de notre travail consiste à proposer une première modélisation PPC de l'extraction de motifs séquentiels sous contraintes – à partir d'une base de séquences – dans un cadre déclaratif permettant de traiter simultanément des contraintes de nature quelconque. Les contraintes traitées dans le cadre de cet article incluent les contraintes de fréquence, clôture, taille, gap et celles portant sur les items (voir (Métivier et al., 2013) pour d'autres types de contraintes traitées et pour une présentation plus détaillée de ce travail). Des expériences me-

TAB. 1: Exemple de base de séquences.

Sequence identifier	Sequence
1	$\langle a\ b\ c\ d\ a\rangle$
2	$\langle d\ a\ e\rangle$
3	$\langle a\ b\ d\ c\ \rangle$
4	$\langle c\ a\rangle$

nées sur la découverte de motifs porteurs de relations entre des gènes et des maladies orphelines, à partir de textes, montrent l'intérêt de notre approche.

2 Extraction de motifs séquentiels

2.1 Motifs séquentiels

Etant donné un ensemble $\mathcal{I}$ de littéraux distincts appelés *items*, une séquence $s = \langle i_1, \ldots, i_n\rangle$ est une liste ordonnée non vide d'items. Une séquence $S_a = \langle a_1, \ldots, a_n\rangle$ est incluse dans une autre séquence $S_b = \langle b_1, \ldots, b_m\rangle$ s'il existe des entiers $1 \leq i_1 < \ldots < i_n \leq m$ tels que $a_1 = b_{i_1}, \ldots, a_n = b_{i_n}$. Si la séquence S_a est incluse dans S_b, alors S_a est une sous-séquence de S_b et S_b est une super-séquence de S_a, noté $S_a \preceq S_b$. Par exemple, la séquence $\langle a\ b\ d\ c\rangle$ est une super-séquence de $\langle b\ c\rangle$: $\langle b\ c\rangle \preceq \langle a\ b\ d\ c\rangle$. Une base de séquences SDB est un ensemble de paires (sid, S) où sid est un identifiant de séquence et S est une séquence. Une paire (sid, S) *contient* une séquence (ou un motif) S_α si S_α est une sous-séquence de S ($S_\alpha \preceq S$). Le support absolu d'une séquence S_α dans une base de séquences SDB correspond au nombre de paires (sid, S) qui contiennent S_α. Le support relatif représente le pourcentage de paires qui supportent S_α ($\frac{|(sid,S)\ t.q.\ S_\alpha \preceq S|}{|SDB|}$). Un motif séquentiel fréquent est un motif ayant un support minimal supérieur ou égal à un certain seuil $minsup$.

2.2 Fouille de motifs séquentiels sous contraintes

Les contraintes permettent à l'utilisateur de définir plus précisément ce qu'il considère comme intéressant pour ne conserver que les motifs pertinents (Dong et Pei, 2007). Un exemple classique de contraintes est celle de support minimal. Nous passons en revue quelques autres contraintes classiques et traitées par la suite :

Fermeture. Cette contrainte permet d'obtenir une représentation condensée des motifs en éliminant les redondances entre motifs : Un motif fréquent s est un motif fermé fréquent, s'il n'existe pas de motif fréquent s' tel que $s \preceq s'$ et $sup(s) = sup(s')$. Par exemple, avec $minsup = 2$, le motif $\langle a\ b\ c\rangle$ de la Table 1 est fermé contrairement au motif $\langle b\ c\rangle$.

Contrainte d'item. Cette contrainte spécifie le sous-ensemble d'items qui doivent apparaître ou non dans les motifs extraits. Par exemple, soit la contrainte $C_{item} \equiv sup(p) \geq 2 \wedge (a \in p) \wedge (b \in p)$, trois motifs séquentiels sont extraits de la table 1 : $\langle a\ b\rangle$, $\langle a\ b\ c\rangle$ et $\langle a\ b\ d\rangle$.

Contrainte de taille. Cette contrainte limite la taille (en nombre d'items) des motifs extraits.

Contrainte de Gap. Un motif séquentiel avec une contrainte de gap $C_{gap} \equiv [M, N]$, notée $p_{[M,N]}$,est un motif tel qu'il y ait au moins M items et au plus N items entre deux items voisins des séquences d'origine. Par exempls, soit $p_{[0,2]} = \langle c\ a\rangle$ et $p_{[1,2]} = \langle c\ a\rangle$ deux motifs avec

deux contraintes de gap différentes, et soient les séquences de la table 1. Les sequences 1 et 4 supportent le motif $p_{[0,2]}$ (la séquence 1 contient un item entre (c) et (a) alors que la séquence 4 ne contient aucun item entre (c) et (a)). Mais seule la séquence 1 supporte $p_{[1,2]}$.

3 Modélisation de la fouille séquentielle sous contraintes

3.1 Programmation par contraintes

La Programmation par Contraintes (PPC) est un paradigme puissant pour résoudre des problèmes combinatoires, se basant sur des techniques issues de l'intelligence artificielle et de la recherche opérationnelle. La PPC se base sur le principe suivant : (1) l'utilisateur spécifie le problème d'une façon déclarative comme un *problème de satisfaction de contraintes* (CSP) ; (2) le solveur cherche l'ensemble complet et correct de solutions du problème. Un CSP est un triplet $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ où $\mathcal{X} = \{X_1, \ldots, X_n\}$ est un ensemble fini de variables ayant pour domaines finis $\mathcal{D} = \{D_1, \ldots, D_n\}$ et $\mathcal{C} = \{C_1, \ldots, C_m\}$ est un ensemble de contraintes où chaque C_i est une condition sur un sous-ensemble de $\mathcal{X}$. L'objectif est de trouver une affectation complète de valeur $d_i \in D_i$ à chaque variable X_i satisfaisant toutes les contraintes de $\mathcal{C}$.

Une technique de modélisation importante en PPC sont les *contraintes globales* qui décrivent un ensemble de propriétés que doit satisfaire un ensemble de variables. Nous présentons succinctement deux contraintes globales, `Among` et `Regular`, permettant de modéliser les contraintes décrites en Section 2.

La contrainte Among. Cette contrainte restreint le nombre d'occurrences de certaines valeurs dans une séquence de n variables (voir (Beldiceanu et Contejean, 1994) pour plus de détails).

La contrainte Regular. Soit M un automate fini déterministe et $\mathcal{X}$ un ensemble de variables, la contrainte $\texttt{Regular}(X, M)$ impose que la séquence de valeurs de $\mathcal{X}$ appartient au langage régulier reconnu par M (Pesant, 2004).

3.2 Modèle

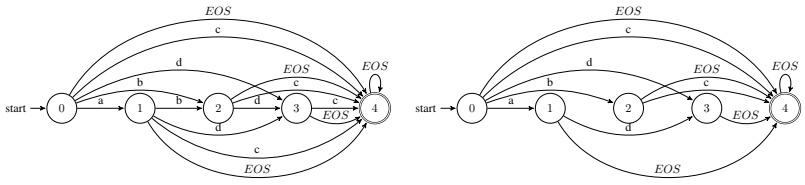
Variables. Soit $I = \{i_1, \ldots, i_n\}$ un ensemble de n items, EOS un symbole n'appartenant pas à I désignant la fin d'une séquence, SDB un ensemble de m séquences et ℓ la taille de la plus grande séquence de SDB. Un motif séquentiel inconnu p de taille ℓ est modélisé par les variables $P_1, P_2, \ldots P_\ell$, chaque P_i a pour domaine $D_i = I \cup \{EOS\}$. On introduit les m variables booléennes T_s telle que ($T_s = 1$) ssi (p est une sous-séquence de s) : $(S_s = 1) \Leftrightarrow (p \preceq s)$. Alors, $\text{sup}(p) = \Sigma_{s \in SDB}\, T_s$.

Modélisation de "$p \preceq s$". Pour chaque séquence s, nous générons un automate A_s permettant de capturer toutes les sous-séquences de s. Ensuite, nous imposons la contrainte `Regular` indiquant que le motif p doit être reconnu par l'automate A_s. Pour réduire le nombre d'états de A_s, pour chaque séquence s, nous considérons uniquement ses items fréquents dans SDB. La figure 1a montre un exemple d'automate généré pour la troisième séquence de la Table 1.

Modélisation de l'extraction de motifs séquentiels. Soit $minsup$ un seuil minimal de fréquence. Ce problème est modélisé par les contraintes suivantes :

$$\forall s \in SDB : T_s = 1 \leftrightarrow \texttt{Regular}(p, A_s) \quad (1)$$

$$sup(p) = \Sigma_{s \in SDB}\, T_s \geq minsup \quad (2)$$



(a) Sans la contrainte de gap. (b) Avec la contrainte de gap (1,1).

FIG. 1: Exemple d'automate pour la séquence $\langle a\ b\ d\ c \rangle$.

3.3 Modélisation des contraintes définies par l'utlisateur

Contrainte d'item. Pour spécifier qu'un sous-ensemble d'items de V doivent être présents au moins une fois dans le motif p, il suffit d'imposer la contrainte $\texttt{Among}(p, V, [l, u])$, avec $0 \leq l \leq u \leq \ell$.

Contrainte de longueur. Les contraintes de longueur minimale et maximale peuvent être modélisées comme suit :

- $len(p) \geq k : \forall i \in [1 \dots k] : P_i \neq EOS$.
- $len(p) \leq k : \forall i \in [k+1 \dots \ell] : P_i = EOS$.

Contrainte de fermeture. Les motif fermés sont les motifs maximaux des classes d'équivalence des motifs partageant la même fréquence. Dans notre modélisation, un motif maximal est un motif ayant le plus petit nombre de variables P_i instanciées à EOS. Ce problème est formulé sous forme d'une *contrainte de minimisation* sur la taille des motif : (1) pour chaque variable P_i, nous posons une fonction de coût unaire c_i tel que $c_i = 1$ si P_i= EOS ; 0 sinon ; (2) minimiser la fonction $c(p) = \sum_{P_i \in p} c_i$. Ainsi, le calcul des fermés se ramène à une contrainte de minimisation sur la taille des motifs et sur l'ensemble de tous les motifs fréquents (contraintes 1 et 2). Enfin, à chaque fois qu'un motif fréquent est prouvé fermé, une contrainte est ajoutée dynamiquement pour interdire de redécouvrir une nouvelle fois ce motif.

Contrainte de Gap. Pour modéliser la contrainte gap $[M, N]$, il suffit de modifier la construction de l'automate A_s de telle sorte à ne garder que les transitions respectant la contrainte de gap. Soit $A_s^{[M,N]}$ le nouvel automate ainsi obtenu. La contrainte réifiée (1) est alors réécrite comme suit : $\forall s \in SDB : T_s = 1 \leftrightarrow \texttt{Regular}(p, A_s^{[M,N]})$. La figure 1b montre le nouvel automate obtenu à partir de celui de la figure 1a avec un gap $[1, 1]$.

4 Éxperimentations

Des expérimentations ont été réalisées dans le cadre d'une application de fouille de textes visant à découvrir des relations entre des gènes et des maladies rares (MR) dans les textes biomédicaux. La fouille de séquences a pour objectif d'extraire des motifs séquentiels utilisés comme patrons linguistiques. Cette application est détaillée dans (Béchet et al., 2012).

A) Protocole expérimental. À partir de la base PubMed, nous avons constitué un corpus de 17 527 phrases contenant toutes au moins un nom de gène et de MR. Ce corpus est la base de séquences qui sera fouillée, et où chaque séquence représente une phrase, et chaque item un

mot de la phrase. Chaque mot est lemmatisé, à l'exception des nom de gène et de MR qui sont remplacés par le token générique GENE ou DISEASE. Afin de découvrir des motifs porteurs de relation linguistique gène – MR, nous avons utilisé et testé avec notre approche PPC les contraintes suivantes :
– *Support minimal.* Trois valeurs ont été testées : 2%, 5% et 10%.
– *Longueur minimale.* Afin d'éliminer les motifs trop petits, et non pertinents linguistiquement, nous fixons ce seuil de longueur à 3.
– *Contrainte d'item.* Au regard des relations linguistiques gène – MR que nous voulons extraire, nous contraignons les motifs à contenir au moins un nom de gène, un nom de MR et un verbe (ou un nom) qui représente le prédicat linguistique [1].
– *Fermeture.* Cette contrainte permet d'éliminer les motifs redondants et de réduire la sortie.

Nous avons testé différentes tailles de corpus (de 50 à 500 phrases). Nous rapportons le nombre de motifs fermés extraits et les temps CPU (en secondes) pour les extraire. Nous indiquons entre parenthèses le nombre de motifs extraits lorsque la résolution n'a pas terminé au bout de 10 heures de calcul. Toutes les expériences ont été menées sur un processeur AMD Opteron 2, 1 GHz et une mémoire vive de 256 GO, en utilisant la bibliothèque toulbar2 [2].

B) Résultats. Des résultats de la table 2, nous pouvons dresser les remarques suivantes.
i) Correction et complétude. Notre approche calcule l'ensemble correct et complet de motifs séquentiels. Nous avons comparé les motifs séquentiels extraits par notre approche avec ceux trouvés par (Béchet et al., 2012), et les deux approches renvoient le même ensemble de motifs.
ii) Pertinence des motifs extraits. Notre approche a permis d'extraire plusieurs *motifs linguistiques* pertinents. De tels motifs permettent de mettre en évidence l'expression des relations linguistiques entre gène et MR, comme par exemple, ces deux motifs traduisant une notion de *causalité* : $\langle(DISEASE)\ (be)\ (cause)\ (by)\ (mutation)\ (in)\ (the)\ (GENE)\rangle$ et
$\langle(DISEASE)\ (be)\ (dominant)\ (frequently)\ (cause)\ (by)\ (GENE)(gene)\rangle$.
iii) Temps CPU. Le temps d'exécution de notre approche augmente en fonction de la taille du corpus. Toutefois, pour les corpus de grande taille (≥ 200) et pour des valeurs de $minsup \leq 2\%$, notre approche ne parvient pas à terminer l'extraction de tous les motifs fermés dans un délai de 10 heures. En effet, l'espace de recherche augmente drastiquement et le solveur passe beaucoup plus de temps pour trouver la première solution. Enfin, en raison du caractère générique de notre approche, il est très difficile de rivaliser et de se comparer avec les meilleurs algorithmes de fouille développés pour quelques contraintes. À l'opposé, nous pouvons combiner de manière très élégante et déclarative plusieurs contraintes de nature diverse, ce qui constitue un point important pour l'extraction de motifs pertinents.

5 Conclusion

Nous avons proposé dans cet article une nouvelle approche croisant des techniques de programmation par contraintes et de fouille pour l'extraction de motifs séquentiels. Notre modèle offre un cadre générique et déclaratif pour modéliser et résoudre des contraintes de nature hétérogène. La faisabilité de notre approche a été mis en évidence par des expériences sur une étude de cas pour la découverte de relations gène-MR à partir d'articles PubMed.

1. Pour chaque mot, sa catégorie grammaticale est stockée dans un lexique.
2. `https://mulcyber.toulouse.inra.fr/projects/toulbar2`.

#sentences	50		100		150		200		250	
	#sol.	temps	#sol.	temps	#sol.	temps	#sol.	temps	#sol.	temps
freq > 2%	129	1,105	329	12,761	441	35,164	(89)	–	(34)	–
freq > 5%	47	285	67	1,571	81	2,091	94	4,119	119	8,516
freq > 10%	4	53	21	251	26	577	29	1,423	28	2,764
#sentences	300		350		400		450		500	
	#sol.	temps	#sol.	temps	#sol.	temps	#sol.	temps	#sol.	temps
freq > 2%	(129)	–	(45)	–	(10)	–	(1)	–	(0)	–
freq > 5%	101	9,620	93	16057	83	21,764	84	35,962	(26)	–
freq > 10%	30	5147	24	4,493	23	7,026	20	13,744	21	17,708

TAB. 2: Nombre de motifs fermés extraits pour différentes tailles de corpus.

Dans ce travail, nous nous sommes restreints aux motifs séquentiels d'items. La modélisation PPC est loin d'être triviale et il s'agit d'une première contribution dans ce cadre, l'extension aux séquences d'itemsets étant une perspective naturelle et à court terme.

Remerciements. Ce travail a été soutenu par l'Agence Nationale de la Recherche, projets Ficolofo ANR-10-BLA-0214 et Hybride ANR-11-BS02-002.

Références

Agrawal, R. et R. Srikant (1995). Mining sequential patterns. In *ICDE'95*.

Béchet, N., P. Cellier, T. Charnois, et B. Crémilleux (2012). Sequential pattern mining to discover relations between genes and rare diseases. In *CBMS'12*.

Beldiceanu, N. et E. Contejean (1994). Introducing global constraints in CHIP. *Journal of Mathematical and Computer Modelling 20*(12), 97–123.

Coquery, E., S. Jabbour, L. Saïs, et Y. Salhi (2012). A sat-based approach for discovering frequent, closed and maximal patterns in a sequence. In *ECAI'12*.

Dong, G. et J. Pei (2007). *Sequence Data Mining*, Volume 33 of *Advances in Database Systems*. Kluwer.

Guns, T., S. Nijssen, et L. D. Raedt (2011). Itemset mining : A constraint programming perspective. *Artif. Intell. 175*(12-13), 1951–1983.

Khiari, M., P. Boizumault, et B. Crémilleux (2010). Constraint programming for mining n-ary patterns. In *CP'10*.

Métivier, J.-P., S. Loudni, et T. Charnois (2013). A constraint programming approach for mining sequential patterns in a sequence database. In *ECML/PKDD Workshop on Languages for Data Mining and Machine Learning*.

Pesant, G. (2004). A regular language membership constraint for finite sequences of variables. In *CP'04*.

Summary

We propose in this paper a Constraint Programming (CP) approach to model and mine sequential patterns in a sequence database. Our CP approach offers a natural way to simultaneously combine in a same framework a large set of constraints coming from various origins.