Extraction de Motifs d'Intervalles Fermés en utilisant la Programmation Par Contraintes

Djawad Bekkoucha, Abdelkader Ouali, Justine Reynaud, Bruno Crémilleux, Patrice Boizumault, Aymeric Beauchamp

Extraction de Motifs d'Intervalles Fermés en utilisant la Programmation Par Contraintes

D. Bekkoucha[1], A. Ouali[1], J. Reynaud[1], B. Crémilleux[1], P. Boizumault[1], A. Beauchamp[2]

[1] Normandie Univ, UNICAEN, ENSICAEN, CNRS, GREYC, 14000 Caen, France
[2] Université d'Orléans, LIFO

[1]prénom.nom@unicaen.fr
[2]prénom.nom@univ-orleans.fr

Résumé

De nombreuses approches en programmation par contraintes (PPC) ont été proposées pour extraire des motifs à partir des données binaires. Ces approches nécessitent une étape préalable de discrétisation binaire pour extraire des motifs sur des données numériques. Cette transformation entraîne souvent une perte d'information ou d'efficacité. A contrario, nous proposons une approche déclarative en définissant une modélisation PPC afin d'extraire directement des motifs d'intervalles fermés à partir des données numériques. La binarisation de ces données est effectuée à la volée afin de préserver l'information originelle. De plus, le modèle proposé peut facilement être étendu avec les modèles existants afin d'extraire des motifs hétérogènes. Les expérimentations menées sur différents jeux de données montrent l'efficacité de notre approche par rapport aux approches déclaratives et sa capacité à découvrir des motifs intéressants.

Mots-clés

Motifs d'intervalles fermés, Programmation par contraintes, Structures de patrons, Motifs hétérogènes.

Abstract

Recent years have seen the rise of constraint programming (CP) to address pattern mining tasks. To deal with numerical data, many of the approaches require a prior discretization of the attributes by using a scaling procedure. Transforming numerical attributes into binary ones leads either to a loss of information or of efficiency. By contrast, we propose to mine on the fly closed interval patterns directly from numerical data in a declarative and efficient way without a binarization step beforehand. Our approach takes advantage of the general CP framework to directly model and mine Closed Interval Patterns from numerical data. Interestingly, we show how the CP model that we define can be easily extended to other data mining tasks such as mining heterogeneous patterns. This is possible by combining our model with other CP models. Experiments conducted on various datasets demonstrate the computational effectiveness of our proposition and its ability to discover meaningful patterns.

Keywords

Closed Interval Patterns, Constraint Programming, Pattern Structures, Heterogenous patterns.

1 Introduction

L'extraction de motifs à partir de données numériques reste un défi majeur, même si différentes approches existent [18, 17] la plupart des approches reposent sur une étape de discrétisation binaire (binarisation) préalable à l'extraction des motifs. Le principe est de diviser l'ensemble des valeurs d'un attribut d'un jeu de données en plusieurs intervalles de valeurs appelés *bins*. Ce processus peut être réalisé à l'aide de techniques supervisées ou non-supervisées qui exploitent des informations telles que des classes, des tests statistiques, l'entropie, etc. [4] ou suivant une approche bayésienne [1]. Les *bins* sont ensuite utilisés comme attributs binaires (i.e. des items) sur l'ensemble des objets pour définir et extraire des motifs. Malheureusement, ces techniques sont susceptibles d'engendrer une perte d'information due à la discrétisation des données numériques. Pour pallier ce problème, Kaytoue et al. [8] ont proposé `MinIntChange`, une méthode dédiée à l'extraction directe (i.e. sans discrétisation) de motifs d'intervalles fermés. Les motifs extraits du jeu de données binaires construit par `MinIntChange` couvrent tous les motifs d'intervalles possibles du jeu de données numériques d'origine. Cependant `MinIntChange` considère uniquement l'extraction de motifs d'intervalles de motifs numériques alors que l'aspect déclaratif de notre méthode permet de traiter d'autres types de motifs comme les motifs hétérogènes.

Cet article porte sur la tâche générale d'extraction de motifs dans des données numériques afin de découvrir de nouvelles connaissances. Notre travail est régi par les deux objectifs suivants. Premièrement, nous voulons conserver l'ensemble de l'information originelle exprimée par les données numériques. Deuxièmement, nous nous plaçons dans un cadre déclaratif et plus précisément dans celui de la Programmation Par Contraintes (PPC) afin de concevoir une méthode capable de s'adapter aux différentes variantes de cette tâche générale. Le paradigme déclaratif permet de spécifier des tâches de fouille de motifs de manière

simple en utilisant des primitives générales de satisfaction de contraintes [10, 7]. Il est plus facile de combiner ces primitives (et d'en incorporer de nouvelles) pour écrire des requêtes traitant de tâches de fouille complexes. Suivre ces deux principes conduit à bénéficier du paradigme déclaratif pour traiter des données numériques.

Dans cet article, nous présentons une nouvelle méthode déclarative pour découvrir des motifs à partir de données numériques, appelée `CP4CIP` (Constraint Programming For Closed Interval Patterns). Nous définissons un modèle PPC pour exprimer les motifs d'intervalles. Nous démontrons comment les motifs d'intervalles fermés peuvent être directement extraits des données numériques avec ce modèle. Une idée clé de notre proposition est d'effectuer le processus de binarisation à la volée selon les valeurs des données. Cela permet de préserver l'information originelle des données numériques.

À notre connaissance `CP4CIP` est la première méthode déclarative capable de découvrir des motifs directement à partir de données numériques. Un autre point important est que `CP4CIP` peut être facilement étendu pour découvrir d'autres types de motifs y compris à partir de données hétérogènes. À titre d'illustration, nous montrons comment une simple combinaison de `CP4CIP` et de `CP4IM` [15] (un modèle PPC reconnu pour extraire des motifs ensemblistes sur les données binaires) permet d'exploiter efficacement des motifs hétérogènes combinant des données numériques et binaires. L'extraction de tels motifs revient à considérer conjointement un contexte numérique et un contexte binaire. Cela illustre la flexibilité de la PPC pour traiter différents types de motifs à partir de données mixtes.

Ce article est organisé comme suit. La section 2 présente les travaux connexes tandis que la section 3 rappelle les notions de base en fouille de données numériques et en programmation par contraintes. La section 4 détaille nos modèles en PPC pour extraire des motifs d'intervalles fermés et des motifs hétérogènes. La section 5 décrit les expérimentations et reporte les résultats. Enfin, nous concluons l'article dans la section 6.

2 Travaux connexes

La fouille de motifs dans des données numériques a commencé par l'extraction des règles d'association quantitatives [18, 17] et différents travaux se sont succédés depuis [16]. Beaucoup d'entre eux sont fondés sur une approche en deux étapes où chaque attribut numérique est discrétisé selon certaines fonctions d'intérêt, comme par exemple le support ou les valeurs de classe. Ensuite, des motifs sont extraits des données discrétisées. Cette approche est univariée (un attribut est discrétisé sans prendre en compte les autres attributs) et conduit à une perte d'information. Toujours dans le domaine des règles d'association, la méthode `QuantMiner` [16] utilise un algorithme génétique pour découvrir de bons intervalles pour les attributs numériques dans les règles d'association en optimisant à la fois le support et la confiance.

Les tuiles et les sous-groupes sont des types de motifs très communs en fouille de données. Kontonasios et al. [9] utilisent un modèle d'entropie maximale pour extraire d'une manière itérative les tuiles intéressantes en fonction d'un modèle de fond. Cependant, ce modèle requiert des valeurs attendues suivant certaines connaissances préalables alors que, dans notre travail, nous considérons des valeurs absolues par rapport à des seuils donnés. Dans le domaine de la découverte de sous-groupes, Nguyen et Vreeken [14] proposent un processus de discrétisation dont l'objectif est de maximiser la qualité moyenne des motifs. Dans [11, 19], une approche basée sur la longueur minimale de description est utilisée pour découvrir des motifs utiles et renvoyer un ensemble de motifs non redondants se chevauchant avec des bornes bien définies. En considérant la notion de motifs à intervalles fermés, la méthode `OSMIND` [13] trouve des sous-groupes optimaux suivant une mesure d'intérêt dans des données purement numériques. Meeng et Knobbe [12] effectuent une comparaison approfondie des méthodes existantes pour traiter les attributs numériques dans les sous-groupes. Notre travail tire avantage du principe des intervalles fermés et ne se limite pas aux sous-groupes.

Afin de chercher à la volée des intervalles intéressants sur des données numériques, Kaytoue et al. ont proposé `MinIntChange` [8], une méthode pour extraire d'une manière exhaustive des motifs numériques en utilisant le cadre de l'Analyse Formelle de Concepts (AFC). Cependant cette méthode est dédiée aux motifs d'intervalles fermés. À notre connaissance, il n'existe pas de méthode déclarative pour découvrir des motifs directement à partir de données numériques. En revanche, différentes approches fondées sur la PPC ont été proposées pour extraire des motifs dans un contexte binaire. Pour comparer `CP4CIP` avec l'état de l'art des méthodes déclaratives, nous utilisons `CP4IM` [15] avec une méthode de binarisation des données numériques sans perte d'information *interordinal scaling* (voir la section 3.1). Le modèle `CP4CIP` diffère de `CP4IM` en deux points. Premièrement, la couverture d'un motif d'intervalles candidat est effectuée à la volée pendant le processus de résolution. Deuxièmement, la relation de fermeture est exprimée en utilisant un encodage différent de celui de `CP4IM`. En PPC, certains travaux utilisent des contraintes globales pour capturer les relations cachées entre un ensemble de variables afin d'améliorer l'efficacité. En considérant les motifs fermés, une contrainte globale a été proposée par [10] pour extraire des motifs fermés a partir de données binaires. Dans cet article, nous nous concentrons sur la partie modélisation en utilisant une solution prête à l'emploi pour les tâches d'extraction de motifs numériques.

3 Préliminaires

Une *base de données numérique* $\mathcal{N}$ est définie par un ensemble d'objets $\mathcal{G}$ dans lequel chaque objet est décrit par un ensemble d'attributs $\mathcal{M}$. Chaque attribut $m \in \mathcal{M}$ possède un domaine $\mathcal{W}_m$ de cardinalité finie contenant toutes les valeurs réelles possibles de m dans $\mathcal{N}$. Le tableau 1 montre un exemple jouet d'un ensemble de

données numériques contenant 5 objets et 3 attributs. Dans ce cadre, une base de données binaires est un cas particulier de base de données numériques ou les valeurs de tous les attributs sont binaires : $\mathcal{W}_m = \{0, 1\}, \forall m \in \mathcal{M}$.

	m_1	m_2	m_3
g_1	182	74	74
g_2	176	99	74
g_3	167	73	28
g_4	190	74	76
g_5	153	76	52

TABLE 1 – Un exemple de base de données numériques.

	Vue 1		Vue 2		
	m1	m2	x	y	z
1	3	5			
2	3	5	×		
3	4	4	×	×	

TABLE 2 – Contexte formel hétérogène composé de deux vues.

Dans les bases de données numériques, un motif est généralement représenté par une conjonction d'intervalles pour restreindre les valeurs sur les attributs numériques. Dans cet article, nous utilisons la notion de motifs d'intervalles comme définie dans [8]. Un *motif d'intervalle* est défini comme un vecteur d'intervalles $\mathcal{I} = \langle [\underline{w_m}, \overline{w_m}] \rangle_{\forall m \in \mathcal{M}}$, ou $\underline{w_m}, \overline{w_m} \in \mathcal{W}_m$. Chaque dimension du vecteur $\mathcal{I}$ correspond à un attribut suivant un ordre canonique sur l'ensemble des attributs $\mathcal{M}$. Un objet g est dans la **couverture** d'un motif d'intervalles $\mathcal{I}$ quand $w_{g,m} \in [\underline{w_m}, \overline{w_m}]$, $\forall m \in \mathcal{M}$. La **fréquence** de $\mathcal{I}$, dénotée par $freq(\mathcal{I})$, est le cardinal de la couverture de $\mathcal{I}$.

Par exemple, $\mathcal{I} = \langle [176, 190], [73, 99], [74, 76] \rangle$ est un motif d'intervalles couvrant les objets $\{g_1, g_2, g_4\}$ dans le tableau 1, avec $freq(\mathcal{I}) = 3$.

3.1 Fouille de motifs d'intervalles fermés

L'analyse formelle de concepts (AFC) [5] est une théorie appliquée qui se concentre sur l'énumération de collections compactes de motifs fermés afin d'éviter la redondance dans les données binaires. Un contexte formel est défini comme un triplet $(\mathcal{G}, \mathcal{M}, \mathcal{R})$, où $\mathcal{R} \subseteq \mathcal{G} \times \mathcal{M}$ est une relation binaire entre $\mathcal{G}$ et $\mathcal{M}$. On utilise $g \; \mathcal{R} \; m$ pour exprimer qu'un objet $g \in \mathcal{G}$ est en relation $\mathcal{R}$ avec un attribut $m \in \mathcal{M}$.

Soit $M \subseteq \mathcal{M}$, un opérateur d'extension $ext(M) = \{g \in \mathcal{G} | \forall m \in M, g \; \mathcal{R} \; m\}$ est l'ensemble des objets contenant tous les éléments $\mathcal{M}$. Soit $G \subseteq \mathcal{G}$, un opérateur d'intention $int(G) = \{m \in \mathcal{M} | \forall g \in G, g \; \mathcal{R} \; m\}$ est l'ensemble des éléments contenus dans tous les objets G.

Une paire sous la forme $(G = ext(M), M = int(G))$ est un concept formel, *i.e.* un motif fermé.

Interordinal Scaling est une méthode utilisée pour binariser des bases de données numériques afin d'extraire des motifs fermés du contexte binaire résultant, ces motifs sont interprétés comme des motifs d'intervalles fermés dans une seconde étape. Cette approche préserve toute l'information contenue dans les données numériques, en créant des paires d'attributs sous la forme suivante : $m \leq w_{g,m}; m \geq w_{g,m}, \forall m \in \mathcal{M}, \forall g \in \mathcal{G}$. Chaque élément de ces paires est utilisé comme un attribut binaire sur l'ensemble des objets. La valeur de l'attribut sur chaque objet est fixée à 1 si la condition est remplie, sinon la valeur est fixée à 0. Malgré la préservation complète de l'information, cette approche produit un énorme ensemble de données ayant un nombre d'attributs s'élevant à $\sum_{m \in \mathcal{M}} 2|\mathcal{W}_m|$. Dans notre travail, nous nous appuyons sur les notions issues des représentations condensées de motifs [2] telles que les motifs fermés pour calculer efficacement ces motifs qui résument l'ensemble de l'information.

Les structures de motifs d'intervalles [6] sont une généralisation de l'AFC pour traiter des données plus complexes telles que des données numériques, des graphes, des séquences, etc. Une structure de motifs est formellement définie comme un triplet $(\mathcal{G}, (D, \sqcap), \delta)$, où $(D, \sqcap)$ est l'inf-demi-treillis de descriptions donné dans un espace d'attributs de $|\mathcal{M}|$ dimensions et ordonné par l'opérateur de similarité $\sqcap$, et $\delta : \mathcal{G} \rightarrow D$ est une application qui associe chaque objet à sa description. Dans le cadre des structures de motifs d'intervalles, un objet $g \in \mathcal{G}$ est décrit par un vecteur d'intervalles comme détaillé précédemment. Soit $\mathcal{I}_1, \mathcal{I}_2 \in D$ deux motifs d'intervalles tels que $\mathcal{I}_1 = \langle [\underline{v_m^1}, \overline{v_m^1}] \rangle_{\forall m \in \mathcal{M}}$ et $\mathcal{I}_2 = \langle [\underline{v_m^2}, \overline{v_m^2}] \rangle_{\forall m \in \mathcal{M}}$. L'opérateur de similarité $\sqcap$ est appliqué aux descriptions des objets et retourne l'enveloppe convexe définie par :

$$\mathcal{I}_1 \sqcap \mathcal{I}_2 = \langle \; [min(\underline{v_m^1}, \underline{v_m^2}), \quad max(\overline{v_m^1}, \overline{v_m^2})] \; \rangle_{\forall m \in \mathcal{M}}$$

L'intersection de motifs dans l'AFC a les propriétés d'un infimum dans un demi-treillis, d'où l'idée d'ordonner les motifs en fonction de la relation : $\mathcal{I}_1 \sqsubseteq \mathcal{I}_2 \Leftrightarrow \mathcal{I}_1 \sqcap \mathcal{I}_2 = \mathcal{I}_1$. La connexion de Galois entre $(2^{\mathcal{G}}, \subseteq)$ et $(D, \sqcap)$ est définie comme suit :

$$\begin{cases} G^{\square} = \bigsqcap_{g \in G} \delta(g), \text{ pour } G \subseteq \mathcal{G}, & \text{et} \\ d^{\square} = \{g \in \mathcal{G} | d \sqsubseteq \delta(g)\}, \text{ pour } d \in (D, \sqcap) \end{cases}$$

$G^{\square}$ est la description commune à tous les objets de G, et $d^{\square}$ est l'ensemble des objets dont la description est subsumée par d. Un motif d'intervalles fermés, également appelé concept du contexte $(\mathcal{G}, (D, \sqcap), \delta)$ est la paire (G, d) tel que $G^{\square} = d$ et $d^{\square} = G$. Par exemple, $(\{g_1, g_2, g_4, g_5\}, \langle [176, 190], [74, 99], [74, 76] \rangle)$ est un motif d'intervalles fermés dans le tableau 1.

Les structures de motifs hétérogènes [3], sont des structures de motifs permettant de combiner différents types de données. Par exemple, considérons des données binaires ainsi que des données numériques représentées dans le tableau 2. Ensuite, nommons chaque type de données une *vue*. Fondamentalement, chaque vue correspond à un contexte formel ; les concepts hétérogènes résultent

de “l’intersection” des concepts formels de chaque vue. Formellement, étant donné deux structures de motifs $(\mathcal{G}, (D_1, \sqcap^1), \delta_1)$ et $(\mathcal{G}, (D_2, \sqcap^2), \delta_2)$, la structure hétérogène du motif $(\mathcal{G}, (D_1 \times D_2, \sqcap), \delta)$ est définie comme suit :

$$\begin{cases} G^{\square} = (\bigsqcap^1_{g \in G} \delta_1(g), \bigsqcap^2_{g \in G} \delta_2(g)), & \text{pour } G \subseteq \mathcal{G} \\ (d_1, d_2)^{\square} = \{g \in \mathcal{G} | d_1 \sqsubseteq \delta_1(g), d_2 \sqsubseteq \delta_2(g)\}, & \text{pour } d_1 \in (D_1, \sqcap) \text{ et } d_2 \in (D_2, \sqcap) \end{cases}$$

3.2 Programmation par contraintes

Un CSP consiste en un ensemble de variables $X = \{X_1, \ldots, X_n\}$, un ensemble de domaines $\mathcal{D}$ faisant correspondre chaque variable $X_i \in X$ à un ensemble fini de valeurs possibles $\mathcal{D}(X_i)$, et un ensemble de contraintes $\mathcal{C}$ sur X. Une contrainte $c \in \mathcal{C}$ est une relation spécifiant les combinaisons de valeurs autorisées pour ses variables $X(c)$. Une affectation sur un ensemble $Y \subseteq X$ de variables est une correspondance entre les variables de Y et les valeurs de leurs domaines. Une solution est une affectation sur X satisfaisant toutes les contraintes. Les solveurs de PPC utilisent généralement la recherche arborescente par retour arrière pour explorer l’espace de recherche des affectations partielles et tentent de les étendre à des affectations cohérentes dans le but de trouver des solutions. La principale technique utilisée pour accélérer la recherche est la propagation des contraintes par un algorithme de filtrage qui permet de réduire la taille des domaines.

4 Modèle PPC pour l’extraction de motifs d’intervalles fermés

Dans cette section, nous présentons une modélisation PPC permettant d’extraire directement des motifs d’intervalles fermés de données numériques, `CP4CIP` (Constraint Programming For Closed Interval Patterns), qui permet de formuler des requêtes exécutant des tâches de fouille complexes. Dans ce qui suit, nous montrons comment le problème d’extraction de motifs d’intervalles fermés est exprimé sous la forme d’un modèle PPC. L’ensemble des intervalles et l’ensemble des objets d’un motif d’intervalles fermés sont exprimés par les variables et les contraintes décrites dans cette section.

4.1 Motifs d’intervalles

On dénote par $[X_m]$ un intervalle sur un attribut $m \in \mathcal{M}$ tel que $[X_m] = \{w \in \mathcal{W}_m : \underline{X_m} \leq w \leq \overline{X_m}\}$. Dans notre modèle nous définissons deux variables pour chaque attribut : une variable pour la borne inférieure $\underline{X_m}$ et une variable pour la borne supérieure $\overline{X_m}$ où $\mathcal{D}(\underline{X_m}) = \mathcal{D}(\overline{X_m}) = \mathcal{W}_m$. Un motif d’intervalles fermés est trouvé en affectant les variables $\underline{X_m}$ et $\overline{X_m}$ à une valeur dans $\mathcal{W}_m$ tel que $\underline{X_m} \leq \overline{X_m}, \forall m \in \mathcal{M}$.

Exemple. *Le motif* $< [176, 190], [74, 99], [74, 76] >$ *est représenté par l’instanciation* $\{\underline{x_1} = 176, \overline{x_1} = 190, \underline{x_2} = 74, \overline{x_2} = 99, \underline{x_3} = 74, \overline{x_3} = 76\}$.

L’extension d’un motif d’intervalles est exprimée en définissant une variable binaire G_g pour chaque objet $g \in \mathcal{G}$ dans un jeu de données numériques pour chaque objet $g \in \mathcal{G}$, où $\mathcal{D}(G_g) = \{0, 1\}$. La variable G_g est fixée à 1 ssi l’objet g est couvert par le motif. Les variables $\underline{X_m}$, $\overline{X_m}$ définissent le vecteur d’intervalles d’un motif d’intervalles sur chaque attribut $m \in \mathcal{M}$, et les variables G_g son ensemble d’objets correspondant dans $\mathcal{G}$.

4.2 Contraintes

Dans cette sous-section, nous décrivons les contraintes modélisant un motif d’intervalles fermé.

Contraintes de couverture $d^{\square}$. Un objet est couvert par un motif d’intervalles si et seulement si toutes les valeurs de ses attributs se trouvent dans les intervalles. Pour formuler la couverture, nous introduisons dans notre modèle des variables binaires $B_{g,m}$, où $\mathcal{D}(B_{g,m}) = \{0, 1\}$ pour chaque valeur m et chaque objet g dans la base de données tel que la contrainte réifiée (1) est satisfaite.

$$B_{g,m} = 1 \iff \underline{X_m} \leq w_{g,m} \leq \overline{X_m}, \forall m \in \mathcal{M}, \forall g \in \mathcal{G} \quad (1)$$

$$G_g = 1 \iff \sum_{m \in \mathcal{M}} B_{g,m} = |\mathcal{M}|, \forall g \in \mathcal{G} \quad (2)$$

La contrainte (2) utilise les variables $B_{g,m}$ pour imposer la couverture sur chaque objet. Un objet $g \in \mathcal{G}$ est couvert ssi la valeur de chaque attribut m de l’objet g est délimitée par l’intervalle $[X_m]$.

Exemple. *D’après le contexte formel ci-dessus (table 1), l’instanciation des variables de couverture pour le motif* $< [176, 190], [74, 99], [74, 76] >$ *est la suivante :* $\{G_1 = 1, G_2 = 1, G_3 = 0, G_4 = 1, G_5 = 0\}$

Contraintes de fermeture $G^{\square} = d$ et $d^{\square} = G$. La fermeture exige que chaque intervalle associé à chaque attribut contienne toutes les valeurs des objets couverts, tandis que chaque valeur d’un objet non couvert doit être en dehors de l’intervalle. Soit $\mathcal{W}^{\uparrow}_m$ (resp. $\mathcal{W}^{\downarrow}_m$) la valeur maximale (resp. minimale) sur les objets de l’attribut m, la relation de fermeture peut être exprimée dans notre modèle en introduisant les nouvelles variables $\underline{H}_{g,m}$ et $\overline{H}_{g,m}$, où $\mathcal{D}(\underline{H_{g,m}}) = \mathcal{W}_m \cup \{\mathcal{W}^{\uparrow}_m + 1\}$, et $\mathcal{D}(\overline{H_{g,m}}) = \mathcal{W}_m \cup \{\mathcal{W}^{\downarrow}_m - 1\}$. La valeur supérieure $\{\mathcal{W}^{\uparrow}_m + 1\}$ (resp. valeur inférieure $\{\mathcal{W}^{\downarrow}_m - 1\}$) est ajouté dans le domaine de $\underline{H}_{g,m}$ (resp. $\overline{H}_{g,m}$) afin d’éviter de sélectionner le minimum (resp. le maximum) sur les objets non couverts. La fermeture du motif d’intervalles est exprimée par les contraintes réifiées (3-8).

$$\forall g \in G, m \in M, G_g = 1 \implies \underline{H_{g,m}} = w_{g,m} \quad (3)$$

$$\forall g \in G, m \in M, G_g = 0 \implies \underline{H_{g,m}} = \mathcal{W}^{\uparrow}_m + 1 \quad (4)$$

$$\forall g \in G, m \in M, \underline{X_m} = min(\underline{H_{1,m}}, \underline{H_{2,m}}, ..., \underline{H_{|\mathcal{G}|,m}}) \quad (5)$$

Pour trouver la valeur minimale sur les objets couverts, nous introduisons les contraintes (3-5). Nous utilisons pour chaque attribut m une contrainte minimale sur les variables

de l'ensemble $\{\underline{H_{g,m}}, \forall g \in G\}$. Les variables $H_{g,m}$ des objets non couverts ont une valeur supérieure à toutes les valeurs des données. Ainsi, le minimum ne peut pas être sélectionné sur les objets non couverts.

$$\forall g \in G, m \in M, G_g = 1 \implies \overline{H_{g,m}} = w_{g,m} \quad (6)$$

$$\forall g \in G, m \in M, G_g = 0 \implies \overline{H_{g,m}} = \mathcal{W}_m^{\downarrow} - 1 \quad (7)$$

$$\forall g \in G, m \in M, \overline{X_m} = max(\overline{H_{1,m}}, \overline{H_{2,m}}, ..., \overline{H_{|\mathcal{G}|,m}}) \quad (8)$$

De la même façon, pour trouver la valeur maximale sur les objets couverts (6-8), nous utilisons pour chaque attribut m une contrainte maximale sur l'ensemble des variables $\{\overline{H_{g,m}}, \forall g \in G\}$. Les variables $\overline{H_{g,m}}$ des objets non couverts ont une valeur inférieure à toutes les valeurs des données. Ainsi, le maximum ne peut pas être sélectionné parmi les objets non couverts.
Les contraintes (3-5) obligent tous domaines des variables d'intervalle ($\underline{X_m}$ et $\overline{X_m}$) à être fermées sur les valeurs des objets couverts.

Exemple. *Les domaines des variables de fermeture pour l'attribut m_1 de l'objet g_1 sont $D(\underline{H_{1,1}}) = \{182, 191\}$ et $D(\overline{H_{1,1}}) = \{152, 182\}$. Le tableau 3 illustre les valeurs prises par les variables $\underline{H_{g,m}}$ (en haut) et $\overline{H_{g,m}}$ (en bas) pour le motif d'intervalles $<$ $[176, 190], [74, 99], [74, 76] >$.* Le minimum sur les valeurs $\{\underline{H_{1,1}}, \underline{H_{2,1}}, \underline{H_{3,1}}, \underline{H_{4,1}}, \underline{H_{5,1}}\}$ est égale à 176.
Le maximum sur les valeurs $\{\overline{H_{1,1}}, \overline{H_{2,1}}, \overline{H_{3,1}}, \overline{H_{4,1}}, \overline{H_{5,1}}\}$ est égale à 190. Le même calcul tient sur les autres attributs.

	$\underline{H_{g,1}}$	$\underline{H_{g,2}}$	$\underline{H_{g,3}}$
g_1	182	74	74
g_2	176	99	74
g_3	191	100	77
g_4	190	74	76
g_5	191	100	77
	$\overline{H_{g,1}}$	$\overline{H_{g,2}}$	$\overline{H_{g,3}}$
g_1	182	74	74
g_2	176	99	74
g_3	152	72	27
g_4	190	74	76
g_5	152	72	27

TABLE 3 – Matrice des valeurs des variables de fermeture inférieure (haut) et supérieure (bas)

Autres contraintes. Notre modèle PPC peut être étendu en utilisant de nouvelles contraintes sur les intervalles et/ou les objets. Les avantages de notre modèle sont doubles : (i) les contraintes supplémentaires sont directement exprimées sur les intervalles plutôt que sur les binaires où un effort de modélisation plus important est nécessaire. (ii) d'autres contraintes complexes peuvent être introduites sur la couverture des objets. Une extension du modèle PPC actuel est donnée dans la section suivante.

4.3 Extension du modèle pour la fouille de motifs hétérogènes

Dans cette section, nous montrons comment `CP4CIP` peut être étendu de façon naturelle pour extraire des motifs hétérogènes tels que ceux présentés dans la section 3.1. Cette tâche est réalisée en combinant le modèle `CP4IM` [15], qui est restreint aux données binaires, avec notre modèle `CP4CIP`. Pour un jeu de données binaires $\mathcal{B}$ composé d'un ensemble d'objets $\mathcal{G}$ et d'un ensemble d'éléments (attributs binaires) $\mathcal{L}$, la modélisation de `CP4IM` utilise deux ensembles de variables booléennes P et G : (i) Variables des items $\{P_1, P_2, ..., P_{|\mathcal{L}|}\}$, où $P_l = 1$ si et seulement si l'item $l \in \mathcal{L}$ est dans l'ensemble des items recherchés ; (ii) Variables d'objet $\{G_1, G_2, ..., G_{|G|}\}$, où $G_g = 1$ ssi l'objet $g \in \mathcal{G}$ est couvert par l'ensemble d'éléments recherché.

$$G_g = 1 \iff \sum_{l \in \mathcal{L}} P_l(1 - \mathcal{B}_{g,l}) = 0, \quad \forall g \in \mathcal{G}. \quad (9)$$

$$(P_l = 1) \iff \sum_{g \in \mathcal{G}} G_g(1 - \mathcal{B}_{g,l}) = 0, \quad \forall l \in \mathcal{L}. \quad (10)$$

`CP4IM` utilise les contraintes (9) et (10) pour faire en sorte que chaque motif solution soit fermé.
Notre modèle peut incorporer directement les contraintes `CP4IM` en utilisant les variables G_g qui sont liées aux variables $\underline{X_m}$ et $\overline{X_m}$ sur les attributs numériques et aux variables $\overline{P_l}$ sur les attributs binaires.

5 Expérimentations

Dans cette section, nous proposons un protocole expérimental pour évaluer deux aspects de notre approche.
Le premier aspect est l'avantage d'incorporer la binarisation à la volée dans le modèle PPC. À cet égard, nous comparons les performances (en termes de temps CPU) de `CP4CIP` à deux méthodes : (i) `CP4IM`, une méthode basée sur le modèle PPC pour la fouille sur des données binaires [15] en lui appliquant un *interordinal scaling* afin de conserver toute l'information (notons qu'il n'existe pas de concurrent selon le paradigme déclaratif sur des données numériques) ; (ii) les méthodes dédiées `MinIntChange` et `LCM-BIN` après application de la binarisation avec *interordinal scaling*.
Dans un second temps, afin d'illustrer la généricité de notre modèle PPC, nous évaluons la flexibilité de `CP4HP` pour la fouille de motifs hétérogènes. Pour cela, nous considérons une application où nous cherchons à caractériser les variables latentes produites par une analyse ACP sur un ensemble de documents en utilisant leurs mots-clés. `CP4HP` peut-il être utilisé pour extraire des motifs hétérogènes utiles qui pourraient expliquer les variables latentes ?

Jeu de données	Originale	
	#attributs	#objets
AP	5	135
BK	5	96
Cancer	9	116
CH	8	209
LW	10	189
NT	3	130
Yacht	7	308
	Interordinal scaled	
	#IS attributs	densité (%)
AP	1348	50.37
BK	626	50.79
Cancer	1800	50.50
CH	792	51.01
LW	506	51.97
NT	134	52.23
Yacht	644	51.08

TABLE 4 – Caractéristiques des jeux de données

Jeux de données. Nous avons sélectionné plusieurs jeux de données numériques difficiles pour les approches déclaratives et mis a disposition par l'Université de Bilkent [1] et qui ont été utilisés dans [8]. Deux autres jeux de données (Cancer et Yacht) ont également été sélectionnés dans les archives de l'UC Irvine. [2]. Les noms des jeux de données sont indiqués par les abréviations standards utilisées dans la base de données de l'Université Bilkent. Le tableau 4 résume les caractéristiques de chaque ensemble de données. Tous les jeux de données sélectionnés sont de tailles et de types différents. La plupart d'entre eux contiennent des valeurs réelles et l'un d'entre eux des valeurs négatives. Dans le cas où des objets ont des valeurs manquantes, nous supprimons complètement l'objet du jeu de données. Pour traiter efficacement les valeurs réelles dans les jeux de données avec l'API OR-tools, nous avons remplacé chaque valeur réelle par un nombre entier qui préserve l'ordre original.

Protocole expérimental. Nous avons implémenté et résolu les modèles PPC en utilisant directement l'API de OR-tools et son module CP-Solver v9.0 [3] au lieu de passer par un langage de modélisation tel que `MiniZinc`. Toutes les expérimentations ont été menées sur un processeur AMD Opteron 6282SE 2,6 GHz et 512 Go de RAM avec un timeout de 5 heures. Pour chaque jeu de données, nous avons diminué le seuil de fréquence (relative) jusqu'à ce qu'il soit impossible d'extraire tous les motifs d'intervalles fermés dans le temps/mémoire alloué. Le code binaire, bases de données et les résultats sont disponibles ici [4]. Pour les méthodes dédiées, nous avons utilisé le code public de `MinIntChange` et `LCM` v3.

1. `http://funapp.cs.bilkent.edu.tr/DataSets/`
2. `https://archive.ics.uci.edu/ml/datasets.php`
3. `https://github.com/google/or-tools/`
4. `https://github.com/oualiaek/cpforcip.git`

données	Freq. %	Temps CPU (secondes)			
		`CP4IM`	`CP4CIP`	`MinIntChange`	`LCM-BIN`
AP	80	630.41	28.47	5.35	**1.29**
	70	4,804.52	192.84	24.34	**13.12**
	60	14,287.49	559.66	74.70	**55.11**
	50	33,401.62	1,219.80	**135.18**	163.04
	20	TO	5,290.18	**464.30**	1,296.05
BK	80	1,431.73	267.95	15.15	**1.43**
	70	11,599.66	1,728.92	117.03	**13.20**
	60	TO	7,303.11	459.97	**51.25**
	50	TO	18,218.68	1,272.80	**157.22**
	20	TO	TO	5,262.99	**1,806.64**
Cancer	95	129.61	18.47	1.81	**0.06**
	94	361.22	46.17	7.63	**0.13**
	92	4,911.79	479.81	34.65	**1.57**
	90	20,623.85	1,815.01	139.64	**5.11**
	80	TO	TO	TO	**3,637.51**
CH	95	15.81	6.04	0.95	**0.01**
	90	432.80	89.01	10.50	**0.28**
	85	3,435.64	648.77	66.64	**2.35**
	80	TO	2,669.77	215.29	**9.31**
	50	TO	TO	21,278.92	**1,580.62**
LW	80	1,264.93	1,605.48	27.77	**1.55**
	70	10,119.56	9,708.12	162.18	**11.17**
	60	TO	31,266.18	647.13	**45.14**
	50	TO	TO	1,842.20	**144.58**
	20	TO	TO	8,861.99	**1,107.47**
NT	80	0.80	1.81	0.23	**0.01**
	50	6.61	10.81	0.63	**0.03**
	20	26.22	27.26	0.71	**0.16**
	10	40.58	31.85	1.06	**0.19**
	1	60.85	32.77	0.83	**0.35**
Yacht	80	34.07	99.36	0.92	**0.03**
	50	5,644.65	4,562.36	24.33	**3.79**
	40	20,781.77	9,643.05	58.94	**12.86**
	30	TO	18,136.41	125.37	**34.92**
	20	TO	30,097.10	226.56	**97.16**

Table (5) Temps de calcul des quatre méthodes sur les différents jeux de données. Le meilleur temps de calcul pour toutes les méthodes considérées est indiqué en gras. Le meilleur temps CPU pour les méthodes PPC est indiqué sur fond gris.

Extraction de motifs d'intervalles fermés. Dans le tableau 5, nous indiquons le temps CPU nécessaire pour trouver tous les motifs d'intervalles fermés en utilisant différents seuils de fréquence minimale pour chaque jeu de données. Pour les méthodes déclaratives, l'observation générale est que `CP4CIP` surpasse significativement `CP4IM` sur la plupart des jeux de données, en particulier lorsque le jeu de données est grand ou que le seuil de fréquence est bas. En termes de temps CPU où `CP4CIP` surpasse `CP4IM`, nous pouvons noter un facteur d'accélération moyen de 9,37 sur tous les jeux de données. Cependant, nous remarquons que `CP4IM` surpasse `CP4CIP` sur les jeux de données NT et Yacht lorsque le seuil de fréquence est élevé. Si l'on considère le jeu de données NT, on peut voir dans le tableau 4 que le nombre d'attributs binaires après le *interordinal scaling* est relativement faible par rapport aux autres jeux de données. De plus, des valeurs plus élevées des seuils de fréquence combinées à des jeux de données plus petits sont bénéfiques pour `CP4IM` puisqu'il utilise un

plus petit nombre de variables et de contraintes avec moins de solutions candidates. Ces observations illustrent l'avantage d'incorporer la binarisation dans CP4CIP.

Les implémentations de MinIntChange et LCM-BIN étant dans des langages différents (Java pour MinIntChange et C pour LCM-BIN) la comparaison de leurs temps CPU est à relativiser. Les résultats sont fournis pour avoir un aperçu de l'écart entre les méthodes déclaratives et les méthodes dédiées.

À partir de cette évaluation, nous pouvons voir l'intérêt d'utiliser CP4CIP lors de la fouille de jeux de données contenant pour chaque attribut un changement significatif sur les valeurs numériques observées. Plus ce changement est important, plus efficace est CP4CIP comparé a CP4IM. Par conséquent, il est plus efficace d'utiliser les contraintes PPC basées sur le type d'attributs malgré la capacité du modèle CP4CIP à extraire des motifs fermés où un item absent dans un motif est interprété comme [0,0] (l'item n'apparaît pas sur tous les objets couverts) ou [0,1] (l'item apparaît sur certains mais pas tous les objets couverts). Ainsi, il est plus efficace d'étendre CP4CIP en utilisant les contraintes de fermeture de CP4IM puisque les attributs sont déjà binaires. Cela motive l'utilisation de CP4HP qui combine les deux modèles pour exploiter efficacement les attributs hétérogènes.

Extraction de structures de motifs hétérogènes. Afin d'illustrer la généricité et l'intérêt de notre approche, nous considérons une application dans laquelle on cherche à caractériser des variables latentes afin d'expliquer les valeurs qu'elles prennent.

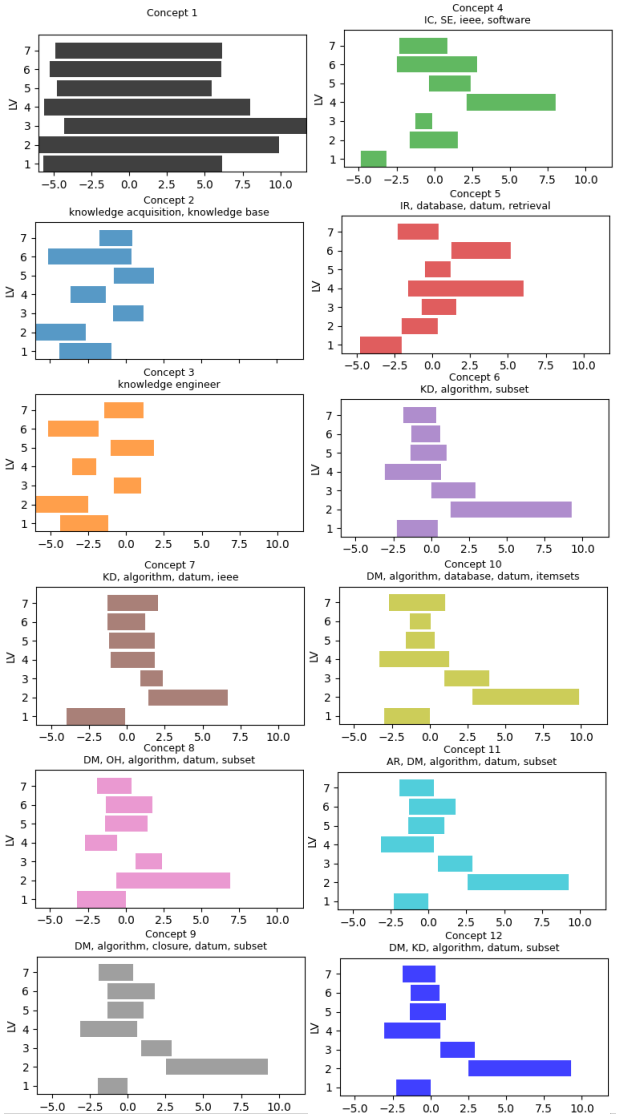
Grâce à des approches comme TF/IDF, nous sommes en mesure de construire une matrice où les lignes correspondent aux documents et les colonnes à un terme. La valeur $0 \leqslant c_{i,j} \leqslant 1$ à la ligne i et à la colonne j dépend de la représentativité du terme t_j pour le document d_i. L'utilisation de l'analyse en composantes principales (ACP) permet de réduire le nombre de dimensions de cette matrice. Chaque nouvelle dimension correspond à une variable latente, connue - dans les informations de recherche - pour représenter un *sujet*.

Malheureusement, ces thèmes ne sont liés à aucune signification réelle. Cependant, en considérant ces vecteurs avec les mots-clés des documents, on peut comprendre le "sens" de certaines parties de l'espace. Par exemple, si tous les documents ayant le mot-clé pattern mining sont dans l'intervalle $[3, 4]$ dans la première dimension, et s'il n'y a aucun autre document dans cet intervalle, on peut supposer que cet intervalle correspond au concept de pattern mining. À cette fin, nous construisons le contexte formel contenant à la fois les variables latentes et les mots-clés. Ensuite, nous profitons de CP4HP pour extraire des concepts hétérogènes. Enfin, ces concepts nous permettent de faire correspondre les mots-clés avec une partie de l'espace latent.

Nous avons utilisé l'API[5] d'ISTEX pour extraire un ensemble de articles de journaux traitant de l'analyse formelle de concepts. Nous avons obtenu 312 articles, ainsi que leurs mots-clés. À partir de ces mots-clés, nous effectuons une ACP afin de représenter chaque article comme un vecteur de 7 dimensions (*i.e.* 7 variables latentes). Ensuite, nous considérons les mots-clés fréquents (*i.e.* apparaissant dans plus de 10 articles, ce qui représente 161 mots-clés) et construisons le contexte hétérogène. Nous avons obtenu un contexte formel hétérogène avec 312 objets, 7 attributs dans la première vue et 161 attributs dans la deuxième vue. Pour chaque attribut de la première vue, le nombre de valeurs uniques est compris entre 304 et 307. La densité de la deuxième vue est de 0.08.

Nous avons utilisé CP4HP et nous avons obtenu des résultats en temps CPU de 186,05 secondes avec 9661 concepts, dont 12 concepts fréquents qui couvrent au moins $3,8\%$ des objets. Ils sont présentés dans la figure 1.

Un rectangle correspond à l'intervalle des valeurs d'une va-



AR : Association Rule ; DM : Datum Mining ; IC : International Conference ; IR : Information Retrieval ; KD : Knowledge Discovery ; OH : Other Hand ; SE : Software Engineering

FIGURE 1 – Concepts hétérogènes fréquents

5. https://api.istex.fr/

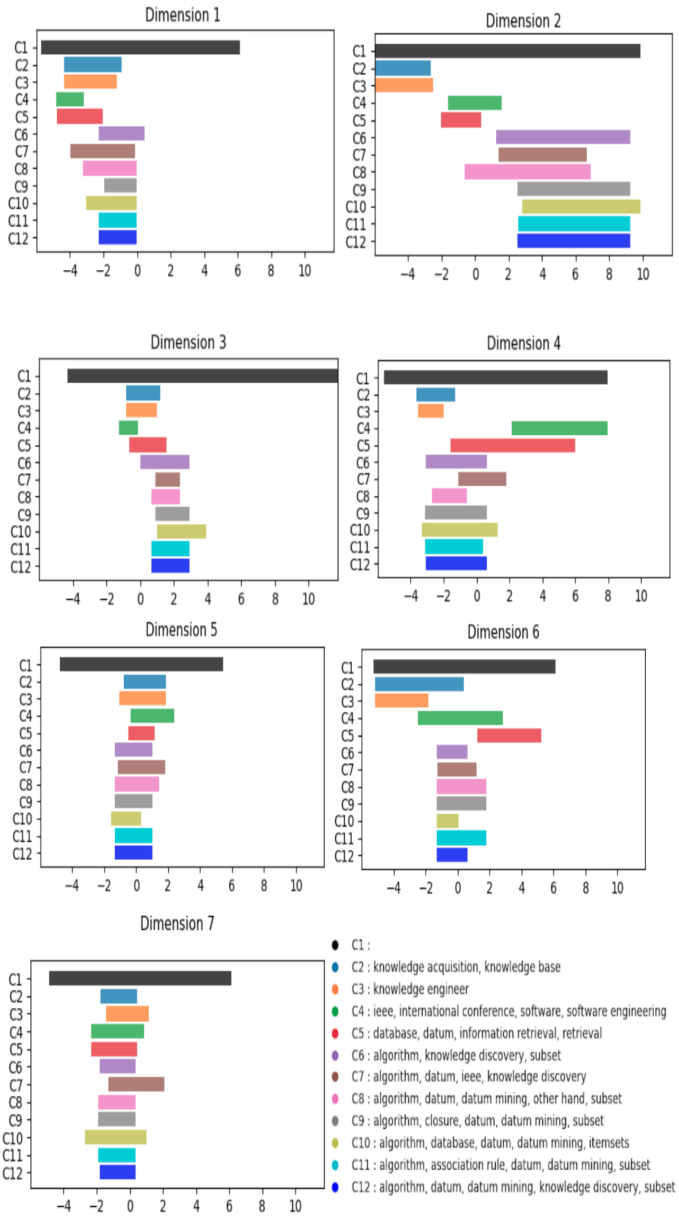


FIGURE 2 – Intervalles de concepts hétérogènes fréquents pour chaque variable latente

riable latente pour ce concept. Par exemple, le concept $C1$ correspond au concept le plus grand : ce concept couvre tous les objets, c'est-à-dire que ses intervalles couvrent toutes les valeurs possibles pour chaque dimension. Il n'y a pas de mot-clé associé à ce concept, car il n'existe pas de mot-clé apparaissant dans tous les articles. La figure fournit un support visuel pour observer les similarités entre les concepts. Avec ces concepts, les similarités dans les valeurs des variables latentes correspondent à des mots-clés similaires. Par exemple, nous observons des similarités entre les concepts qui incluent le mot-clé `algorithme`.

Dans la figure 2, nous comparons les concepts pour chaque dimension (c'est-à-dire le variable latente). Cela nous permet de visualiser la distribution des concepts.

Par exemple, si nous considérons la dimension 5, nous observons que tous les concepts représentés (sauf le "concept supérieur") sont dans $[-2, 2]$, ce qui signifie que cette dimension n'est pas utile pour distinguer les concepts. En revanche, si nous observons la dimension 6, les concepts C6 à C12 ont leurs intervalles entre $-1,8$ et 2, ce qui n'est le cas d'aucun autre concept. Tous ces concepts ont le mot clé `algorithme`.

Dans la plupart des dimensions nous observons que les concepts fréquents (excepté le concept supérieur) ne couvrent pas la totalité des valeurs observées. Cela signifie que certaines parties de ces intervalles représentent des mots-clés qui n'apparaissent pas dans les concepts fréquents.

6 Conclusion

Cet article présente `CP4CIP`, une méthode déclarative permettant de découvrir des motifs d'intervalles à partir de données numériques sans discrétisation préalable. `CP4CIP` conserve la totalité de l'information originale exprimée par les données numériques en effectuant à la volée le processus de binarisation en fonction des données concrètes. `CP4CIP` tire parti du cadre général de la PPC pour modéliser et exploiter directement les motifs d'intervalles fermés. `CP4CIP` peut être facilement étendu pour découvrir d'autres types de motifs à partir de données variées comme extraire efficacement des motifs hétérogènes mélangeant des données numériques et binaires. Dans le cadre d'un travail futur, nous étudions l'amélioration des modèles PPC actuels afin d'obtenir un meilleur compromis entre flexibilité et efficacité.

7 Remerciements

Le premier auteur est financé par l'ANR et la Région Normandie dans le cadre du projet HAISCoDe.

Références

[1] M. Boullé. MODL : A bayes optimal discretization method for continuous attributes. *Mach. Learn.*, 65(1) :131–165, 2006.

[2] T. Calders, C. Rigotti, and J-F. Boulicaut. A survey on condensed representations for frequent sets. In *Constraint-Based Mining and Inductive Databases*, volume 3848 of *LNCS*, pages 64–80. Springer, 2005.

[3] Victor Codocedo and Amedeo Napoli. A Proposition for Combining Pattern Structures and Relational Concept Analysis. In *ICFCA*, 2014.

[4] J. Dougherty, R. Kohavi, and M. Sahami. Supervised and unsupervised discretization of continuous features. In *Machine learning : Proceeding of the twelfth international conference*, pages 194–202. M. Kaufmann, 1995.

[5] B. Ganter and R. Wille. *FCA - Mathematical Foundations*. Springer, 1999.

[6] Bernhard Ganter and Sergei O. Kuznetsov. Pattern structures and their projections. In *ICCS*, 2001.

[7] A. Hien, S. Loudni, N. Aribi, Y. Lebbah, M. E. A. Laghzaoui, A. Ouali, and A. Zimmermann. A relaxation-based approach for mining diverse closed patterns. In *ECML-PKDD, 2020.* Springer.

[8] Mehdi Kaytoue, Sergei Kuznetsov, and Amedeo Napoli. Revisiting numerical pattern mining with formal concept analysis. *IJCAI*, 2011.

[9] K.-N. Kontonasios, J. Vreeken, and T. De Bie. Maximum entropy models for iteratively identifying subjectively interesting structure in real-valued data. In *ECML/PKDD*, 2013.

[10] N. Lazaar, Y. Lebbah, S. Loudni, M. Maamar, V. Lemière, C. Bessiere, and P. Boizumault. A global constraint for closed frequent pattern mining. In *Int. Conf. on Principles and Practice of Constraint Programming*, 2016.

[11] T. Makhalova, Sergei O. Kuznetsov, and A. Napoli. Mint : MDL-based approach for Mining INTeresting Numerical Pattern Sets. *Data Min. Knowl. Discov.*, 36 :108–145, 2022.

[12] M. Meeng and A. J. Knobbe. For real : a thorough look at numeric attributes in subgroup discovery. *Data Min. Knowl. Discov.*, 35 :158–212, 2021.

[13] A. Millot, R. Cazabet, and J-F. Boulicaut. Optimal subgroup discovery in purely numerical data. In *PAKDD*, 2020.

[14] H. Vu Nguyen and J. Vreeken. Flexibly mining better subgroups. In *SDM*, 2016.

[15] L. De Raedt, T. Guns, and S. Nijssen. Constraint programming for data mining and machine learning. In *AAAI*, 2010.

[16] A. Salleb-Aouissi, C. Vrain, and C. Nortet. Quantminer : A genetic algorithm for mining quantitative association rules. In *IJCAI*, 2007.

[17] C. Song and T. Ge. Discovering and managing quantitative association rules. In *ACM Int. Conf. on CIKM*, 2013.

[18] R. Srikant and R. Agrawal. Mining quantitative association rules in large relational tables. In *ACM/SIGMOD*, 1996.

[19] J. Witteveen, W. Duivesteijn, A. J. Knobbe, and P. Grünwald. Realkrimp - finding hyperintervals that compress with MDL for real-valued data. In *IDA*, 2014.