UNIVERSITÉ DE CAEN BASSE-NORMANDIE
UFR DE SCIENCES
DÉPARTEMENT D'INFORMATIQUE

Mémoire de Master
(avril à septembre 2011)

Relaxation de contraintes pour l'extraction de motifs

Auteur :
Willy Ugarte Rojas

Encadrants :
Patrice Boizumault
Samir Loudni

Master Recherche en Informatique (M2) 2010-2011
Spécialité : Algorithmes et Modeles d'Information (AMI)

26 septembre 2011

Remerciements

Je remercie mes encadrants Patrice Boizumault et Samir Loudni de m'avoir fait confiance, de m'avoir soutenu dans les démarches de l'allocation ministérielle et d'avoir encadré chaque étape de ce travail.

Je remercie en particulier Bruno Crémilleux pour sa collaboration dans les différentes étapes de mon stage.

Je tiens à remercier mes amis de la faculté d'ingénierie informatique à Cusco qui ont toujours été présents malgré l'éloignement.

Je souhaite aussi remercier tout le personnel du GREYC et du département informatique pour leur disponibilité et leur gentillesse.

Finalement, je remercie tous les membres de ma famille : ma mère, mon père et mon frère Héctor qui m'ont toujours soutenu.

Table des matières

Introduction

L'extraction de motifs locaux en fouille de données est un sujet très étudié. Les motifs locaux pertinents sont spécifiés sous forme de contraintes (propriétés qu'ils doivent satisfaire). Les méthodes de recherche mises en œuvre utilisent les propriétés spécifiques des contraintes telles que l' (anti-)monotonicité pour réaliser du filtrage et réduire de manière drastique l'espace de recherche. En revanche, jusqu'à récemment, seules des méthodes ad hoc ont été proposées pour l'extraction de motifs de plus haut niveau tels que les motifs n-aires qui mettent en relation plusieurs motifs. Pour chaque motif n-aire ou règle portant sur plusieurs motifs, il est nécessaire de développer une méthode de recherche spécifique [23, 40]. Un tel coût de mise en œuvre a constitué un frein pour l'utilisation des motifs n-aires.

La Programmation Par Contraintes (PPC) [37] offre un cadre générique pour modéliser et résoudre les problèmes d'extraction de motifs en fouille de données. L'utilisation de la PPC pour l'extraction de motifs est un domaine de recherche très récent. Le projet CP4IM (KU Leuven) a montré l'apport de la PPC pour l'extraction de motifs locaux [22, 34, 35, 36]. En ce qui concerne l'équipe CoDaG, Mehdi Khiari a proposé deux approches utilisant la PPC pour l'extraction de motifs n-aires [17, 18]. Le point commun de l'ensemble de ces travaux est de modéliser les problèmes d'extraction de motifs, qu'ils soient locaux ou n-aires, sous forme de problèmes de satisfaction (CSP). Ainsi, un motif (local ou n-aire) ne sera retenu que s'il vérifie toutes les contraintes.

Mais, l'utilisateur aimerait pouvoir spécifier des préférences entre contraintes et en relaxer certaines lorsque le nombre de motifs retenus devient très petit (ou nul dans les cas extrêmes). Par exemple, pour la recherche d'un ensemble de molécules couvrant un espace chimique, l'utilisateur peut indiquer la taille minimale de l'espace qui doit être couvert, le recouvrement maximal autorisé entre les espaces chimiques de deux molécules intervenant dans la solution, ou encore exiger que certaines types de molécules soient présents ou pas, etc. Cependant, il est possible que, au final, l'utilisateur préfère une solution avec une molécule d'un type non présélectionné mais qui permet de couvrir un grand espace chimique avec peu de molécules.

Nos travaux en cours sur la relaxation de contraintes pour l'extraction de motifs ont pour objectifs de lever les deux verrous suivants :

i) **la rigidité du cadre actuel** qui fait qu'une solution potentiellement intéressante n'est pas prise en compte dès qu'une contrainte et/ou une valeur seuil pour une contrainte sur une mesure n'est pas vérifiée.

   Dans la pratique, on préfère, par exemple, accepter entre deux molécules un recou-

vrement un peu supérieur au maximum demandé si ce choix conduit à une bonne solution globale pour couvrir l'espace chimique. Pour cela, **nous proposons de relaxer certaines contraintes portant sur des mesures** par l'introduction de seuils souples.

ii) **l'impossibilité pour l'utilisateur d'exprimer des préférences entre contraintes**, pour prendre en considération des priorités ou encore des incertitudes. Souvent, certains motifs s'avèrent plus intéressants que d'autres.

Par exemple, on préfère des molécules d'un certain type chimique par rapport à un autre. Ou encore, lors de la construction d'un classifieur pour prédire la toxicité, le biochimiste préfère que certains attributs soient choisis avant d'autres. Il existe relativement peu de travaux en fouille de données pour traduire des relations entre des préférences exprimées par un utilisateur : citons les motifs les plus informatifs traduisant une relation de dominance partiellement locale entre motifs selon une fonction de score [24] ou les skylines qui retournent les points d'intérêts non dominés par les autres critères dans un espace [12].

Ce travail de stage de master propose une nouvelle approche pour lever le premier verrou.

Annonce du plan

Le chapitre 1 rappelle les travaux et concepts existants pour l'extraction de motifs sous contraintes. Le chapitre 2 rappelle les notions de base relatives aux CSPs dont nous aurons besoin dans ce mémoire. Le chapitre 3 présente la relaxation de contraintes et motive notre choix du cadre de la relaxation disjonctive [27] pour la relaxation des contraintes pour l'extraction de motifs.

Dans le chapitre 4, nous étudions la relaxation des contraintes de seuil, où nous proposons trois sémantiques de violation. Pour la première sémantique (écart absolu), le coût de violation est la distance de la mesure du motif au seuil. Pour la seconde sémantique (écart relatif), le coût de violation est la distance de la mesure du motif au seuil normalisé. Pour la troisième sémantique (écart relatif restreint), si on juge que la mesure du motif est trop loin du seuil, alors on considère que la contrainte est insatisfaite donc le coût de violation est infini, sinon le coût de violation est la distance de la mesure du motif au seuil normalisé.

Dans le chapitre 5, nous présentons l'implantation en Gecode [16] pour la mise en œuvre de notre travail. Le chapitre 6 décrit les expérimentations faites sur les règles d'exception [40] et les règles inattendues [23]. Le chapitre 7 présente un schéma de relaxation montrant l'intérêt de notre approche sur une application réelle dans le domaine de la chémoinformatique [29, 33, 28, 30, 32, 31]. Enfin, nous concluons et dressons différentes perspectives prolongeant ce travail.

Chapitre 1

Extraction de motifs

Définition 1 (item)**.** *Un **item** est un littéral que nous noterons en lettres capitales A,B,etc.*

$\mathcal{I}$ désigne l'ensemble des items et est de cardinalité n.

Définition 2 (transaction)**.** *Une **transaction** t est un ensemble d'items, $t \subset \mathcal{I}$ formant une entrée de la base de données. Par exemple :*$\{A,ABC,AEF,DCF,\ldots\}$

$\mathcal{T}$ désigne l'ensemble des transactions et est de cardinalité m.

Soit $(d_{i,j})_{1\leq i\leq n}$ et $_{1\leq j\leq m}$ la matrice booléenne associée à $\mathcal{T}$. Cette matrice est définie par :$d_{i,j} = 1$ ssi l'item i appartient à la transaction j.

Définition 3 (base de données)**.** *Une base de données formelle est la donnée d'un couple $(\mathcal{T},\mathcal{P})$ où :*
- *$\mathcal{T}$ est un ensemble fini de transactions,*
- *$\mathcal{I}$ est un ensemble fini d'items.*

▪ **Exemple 1.** *Considérons par exemple la base de données suivante :*

Trans.	Items				
t_1	A		C	D	
t_2		B	C		E
t_3	A	B	C		E
t_4		B			E
t_5	A	B	C		E
t_6		B	C		E

où :
- $\mathcal{T} = \{t_1, t_2, t_3, t_4, t_5, t_6\}$.
- $\mathcal{I} = \{A, B, C, D, E\}$.

1.1 Extraction de motifs locaux

Définition 4 (motif local)**.** *Un* ***motif local*** *est un sous-ensemble non-vide de $\mathcal{I}$. L'ensemble de tous les motifs d'une base est donc l'ensemble des parties de $\mathcal{I}$, noté $2^{\mathcal{I}}\backslash\emptyset$.*

Définition 5 (taille)**.** *La* ***taille*** *d'un motif est le le nombre d'items que ce motif contient. Soit X un motif local, taille$(X) =\mid \{i \in X\} \mid$ où $i \in \mathcal{I}$.*

▪ **Exemple 2.** *Pour la base de données en exemple 1, on a donc :*

- 5 motifs de taille 1 : {A}, {B}, {C}, {D} et {E}.
- 10 motifs de taille 2 : {AB}, {AC}, {AD}, {AE}, {BC}, {BD}, {BE}, {CD}, {CE} et {DE}.
- 10 motifs de taille 3 : {ABC}, {ABD}, {ABE}, {ACD}, {ACE}, {ADE}, {BCD}, {BCE}, {BDE}, et {CDE}.
- 5 motifs de taille 4 : {ABCD}, {ABCE}, {ABDE}, {ACDE} et {BCDE}.
- 1 motif de taille 5 : {ABCDE}.

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 $\mathcal{L}_{\mathcal{I}}$.

Définition 6 (Extraction de Connaissances des Bases de Données (ECBD))**.** *L'ECBD peut être considérée comme un processus d'extraction de connaissances* nouvelles*, potentiellement* utiles *et ayant un degré de* plausibilité*, dans de* grands volumes *de données.*

- *nouvelles : c'est-à-dire pas déjà connues.*
- *utiles : réutilisable dans un processus de raisonnement.*
- *plausibles : on cherche à contrôler la plausibilité des connaissances extraites.*
- *grands volumes de données :*
  - *nécessite des processus automatiques.*
  - *permet une certaine "validité statistique" des connaissance extraites.*

La recherche de motifs locaux est une tâche centrale en ECBD. Ces motifs peuvent correspondre à des sous-parties des données, éventuellement de faible taille ou impliquant peu d'attributs mais qui ont un fort intérêt parce qu'ils traduisent un comportement qui s'écarte des connaissances générales sur les données. La recherche de motifs locaux est au cœur de l'extraction sous contraintes. Une contrainte permet à l'utilisateur de focaliser la recherche de l'information à extraire suivant ses centres d'intérêts.

L'extraction de motifs a pour but la découverte d'informations à partir de tous les motifs ou d'un sous-ensemble de $\mathcal{L}_{\mathcal{I}}$. L'extraction sous contraintes cherche la collection de tous les motifs de $\mathcal{L}_{\mathcal{I}}$ présents dans $\mathcal{T}$ et satisfaisant un prédicat appelé *contrainte*. Ces motifs sont appelés *motifs locaux*, ce sont des régularités observées dans certaines parties des données. La localité de ces motifs provient du fait que, vérifier s'ils satisfont une contrainte donnée, peut s'effectuer indépendamment des autre motifs. La découverte de motifs sous contraintes a pour but de sélectionner les motifs locaux d'une base de données, qui satisfont une contrainte.

Définition 7 (mesure)**.** *Une* ***mesure*** *est une fonction qui associe une valeur (une longueur, une probabilité, etc) à des sous-ensembles d'un ensemble donné.*

Pour l'ECBD une mesure *est une fonction qui associe une valeur à un motif.*

Définition 8 (couverture)**.** *La* ***couverture*** *d'un motif est l'ensemble de transactions qui le contiennent. Soit X un motif local, $couverture(X) = \{t \in \mathcal{T} \mid X \subset t\}$.*

Si une transaction $t \in couverture(X)$, alors X couvre t.

Définition 9 (fréquence)**.** *La* ***fréquence*** *d'un motif est le nombre de transactions qui le contiennent. Soit X un motif local, $freq(X) =\mid \{t \in \mathcal{T} \mid X \subset t\} \mid=\mid couverture(X) \mid$.*

Une contrainte de *fréquence* permet de sélectionner les motifs qui apparaissent dans la base de données un nombre de fois qui dépasse un seuil minimal fixé par l'utilisateur : $freq(X) \geq minfr$.

malades

Patient	Descripteurs					
P_1	A	B			E	F
P_2	A				E	
P_3	A	B	C	D		
P_4	A	B	C	D	E	
P_5				D	E	
P_6			C			F

TABLE 1.1 – Descripteurs médicaux caractérisant un groupe pathologique.

▪ **Exemple 3.** *Considérons une étude médicale portant sur la maladie de l'athérosclérose et dont le but est d'identifier des facteurs pathogènes*[1] *(Supposons qu'on dispose de données comme celles indiquées dans le tableau 1.1). Le contexte* malades *représente 6 patients identifiés par P_1, ..., P_6 et décrits par les 6 descripteurs étiquetés de A à F. La première ligne signifie que les 4 descripteurs A,B,E et F sont présents pour le patient P_1. Par exemple, le descripteur A correspond à une forte consommation de tabac ; le descripteur B, à des antécédents familiaux ; le descripteur C, à une taille supérieure à 1m80 ;etc. Les medecins sont intéressés par les combinaisons de descripteurs présents auprès de nombreux patients car ceux-ci sont de potentiels facteurs de risque. De telles régularités sont appelées* motifs fréquents*. Plus précisément, un motif est dit fréquent si son nombre de répétitions (ici, le nombre de patients qu'il caractérise) excède un seuil fixé. La fréquence de {A,B}, dénotée* freq ({A,B})*, est 3 car A et B apparaissent simultanément chez le patients P_1, P_3 et P_4. De cette manière, si le seuil minimal retenu est 3, le motif {A,B} (i.e. "une forte consommation de tabac accompagnée d'antécédents familiaux") sera extrait. Ce dernier n'est qu'un exemple de la collection des 7 motifs du contexte satisfaisant la* contrainte $freq(X) \geq 3$*, à savoir {A},{B},{C},{D},{E},{A,B} et {A,E}.*

1. Ces données ont été utilisées lors de plusieurs ECML/PKDD Discovery Challenges

Plus généralement, pour les motifs ensemblistes, les objets d'études formant la base de données sont appelés *transactions* et leurs descripteurs, *items*. Cette terminologie est issue de la tâche originelle de l'analyse du "panier du consommateur" [1]. Dans l'exemple, les transactions modélisent donc les patients et les items, les descripteurs.

Deux autres mesures utilisées fréquemment sont :

Définition 10 (aire). *Soit X a motif local,* ***la mesure d'aire*** *est définie par :*

$$aire(X)=fréquence(X)\times taille(X)$$

Le neuf motifs suivants $\{A,C\}$, $\{B,C\}$, $\{B,E\}$, $\{C,E\}$, $\{A,B,C\}$, $\{A,B,E\}$, $\{A,C,E\}$, $\{B,C,E\}$ et $\{A,B,C,E\}$ satisfont la contrainte $aire(X) \geq 6$.

Pour la mesure suivante on a 2 classes définies par les ensembles de transactions $\mathcal{D}_1, \mathcal{D}_2 \subset \mathcal{T}$.

Définition 11 (émergence). *L'**émergence** ou taux de croissance d'un motif X est définie par :*

$$émergence(X) = \frac{|\mathcal{D}_2|\times freq(X,\mathcal{D}_1)}{|\mathcal{D}_1|\times freq(X,\mathcal{D}_2)}$$

où $\mathcal{D}_1$ et $\mathcal{D}_2$ sont les ensembles de données correspondant aux 2 classes.

1.2 Extraction de motifs n-aires

Définition 12 (contrainte n-aire). *Une **contrainte n-aire** porte sur plusieurs motifs locaux.*

Définition 13 (motif n-aire). *Un **motif** est **n-aire** s'il figure dans une* contrainte n-aire.

Peu de travaux concernant l'extraction de motifs n-aires ont été menés et les méthodes développées sont toutes *ad hoc* [40]. La difficulté de la tâche explique l'absence de méthodes génériques : en effet, si l'extraction de motifs locaux nécessite déjà le parcours d'un espace de recherche très conséquent, celui-ci est encore plus grand pour l'extraction de motifs n-aires (le passage d'un à plusieurs motifs augmente fortement la combinatoire [2]). Ce manque de généricité est un frein à la découverte de motifs pertinents et intéressants car chaque contrainte n-aire entraîne la conception et le développement d'une méthode *ad hoc*.

Les contraintes n-aires permettent de modéliser un large ensemble de motifs utiles à l'utilisateur tel que la découverte de règles d'exception [40] ou les règles inattendues [23].

▪ **Exemple 4.** *Les contraintes n-aires sont une façon naturelle de concevoir des motifs tolérants aux fautes et candidats à être des groupes : ceux-ci sont définis par l'union de plusieurs motifs locaux satisfaisant une contrainte d'aire (cf définition 10) et ayant un fort recouvrement entre eux. Plus précisément, à partir de deux motifs locaux* X *et* Y*, on définit la contrainte binaire suivante :*

2. Nous devons prendre en compte et comparer les solutions satisfaisant chaque motif impliqué dans les contraintes.

$$c(X,Y) \equiv \begin{cases} aire(X) > min_{aire} \\ aire(Y) > min_{aire} \\ aire(X \cap Y) > \alpha \times min_{aire} \end{cases}$$

où min_{aire} est le seuil minimal d'aire et α est un paramètre forni par l'utilisateur pour fixer le recouvrement minimal entre motifs locaux.

Comme cas spécial de motifs n-aires, on peut définir les règles d'association :

Définition 14 (règle d'association)**.** *La relation $A \rightarrow B$ est appelée **règle d'association** entre A et B ($A \neq B$) avec $A \subseteq \mathcal{I}$, $B \subseteq \mathcal{I}$ et $A \cap B = \emptyset$. A est appelée la **prémisse** de la règle et B la **conclusion**. Une règle est caractérisée par son **support** et sa **confiance**.*

Dans ce cas, une *mesure* est une fonction qui associe une valeur réelle à une règle d'association $X \rightarrow Y$. Les mesures utilisées le plus fréquemment sont les suivantes :

Définition 15 (support)**.** *Le **support** ou* taux de couverture *d'une règle d'association $A \rightarrow B$ est défini par :*

$$support(A \rightarrow B) = \frac{freq(A \cup B)}{n}$$

$$où\ n = card(\mathcal{T}).$$

Cette mesure représente le pourcentage d'objets (transactions) vérifiant la règle.

Définition 16 (confiance)**.** *La **confiance** d'une règle d'association $A \rightarrow B$ est définie par :*

$$confiance(A \rightarrow B) = \frac{freq(A \cup B)}{freq(A)}$$

Cette mesure représente le pourcentage d'objets vérifiant la conclusion de la règle parmi ceux qui vérifient la prémisse.

Les requêtes n-aires nous permettent de décrire beaucoup de motifs demandés par l'utilisateur tels que la découverte de paires de règles d'exception [40] ou les règles inattendues [23].

1.2.1 Règle d'exception

Une règle d'exception est définie comme un motif combinant une règle générale et une règle déviationelle par rapport à la règle générale. L'intérêt d'une règle est mis en évidence par la comparaison avec l'autre règle.

La comparaison entre les règles signifie que ces règles d'exception *ne sont pas* des motifs locaux. Cela nous permet de distinguer les règles d'exception des règles rares où une règle rare est une règle ayant une fréquence très faible. Ceci est utile car, en pratique, les règles rares ne peuvent pas être utilisées directement, car beaucoup d'entre elles surviennent par hasard et ne sont pas fiables.

Définition 17 (règle d'exception)**.** *Une* ***règle d'exception*** *est définie dans le contexte d'une paire de règles où I est un élément (par exemple une valeur de classe) et X, Y sont des motifs locaux :*

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

Une telle paire de règles est composée d'une règle générale $X \backslash Y \to I$ et d'une règle déviationelle $X \to \neg I$. La règle déviationelle isole information inattendue. Cette définition suppose que la règle générale a une fréquence élevée et une confiance assez élevée et la règle déviationelle a une fréquence faible et un degré de confiance très élevé.

Soient $minfr, maxfr, \delta_1, \delta_2 \in \mathbb{N}$ des seuils. La requête n-aire modélisant une règle d'exception est formulée comme suit :

$X \backslash Y \to I$ doit être une règle fréquente ayant une valeur de confiance élevée : $\rightarrow$ $freq((X \setminus Y) \sqcup I) \geq minfr \wedge (freq(X \setminus Y) - freq((X \setminus Y) \sqcup I)) \leq \delta_1$

$X \to \neg I$ doit être une règle rare ayant une valeur de confiance très élevée : $\rightarrow$ $freq(X \sqcup \neg I) \leq maxfr \wedge freq(X) - freq(X \sqcup \neg I) \leq \delta_2$

Pour résumer :

$$exception(X, Y) \equiv \begin{cases} freq((X \setminus Y) \sqcup I) \geq minfr \wedge \\ freq(X \setminus Y) - freq((X \setminus Y) \sqcup I)) \leq \delta_1 \\ \wedge \\ freq(X \sqcup \neg I) \leq maxfr \wedge \\ freq(X) - freq(X \sqcup \neg I) \leq \delta_2 \end{cases}$$

Dans notre exemple (voir Tableau 1.2), la règle de $AC \to \neg c_1$ est une règle d'exception parce que nous avons conjointement une règle générale $A \to c_1$ et une règle déviationelle $AC \to \neg c_1$. Notez que Suzuki propose une méthode fondée sur une estimation probabiliste [40] pour extraire les règles d'exception, mais cette méthode est *ad hoc*.

1.2.2 Règle inattendue

Padmanabhan et Tuzhilin ont introduit dans [23] la notion de *règle inattendue* $X \to Y$ par rapport à une croyance $U \to V$ où U et V sont des motifs. Une règle inattendue est définie dans [23] par :

1. $Y \wedge V$ n'est pas valide,
2. $X \wedge U$ est valide (XU est un motif fréquent),

Trans.	Items
t_1	A B c_1
t_2	A B c_1
t_3	C c_1
t_4	C c_1
t_5	C c_1
t_6	A B C D c_2
t_7	C D c_2
t_8	C c_2
t_9	D c_2

TABLE 1.2 – Exemple de contexte transactionnel $\mathcal{T}$.

3. $XU \rightarrow Y$ est valide ($XU \rightarrow Y$ est une règle fréquente et de confiance suffisante),
4. $XU \rightarrow V$ n'est pas valide (soit $XU \rightarrow V$ n'est pas une règle fréquente, soit $XU \rightarrow V$ est une règle de faible confiance),

Définition 18 (règle inattendue)**.** *Une* ***règle inattendue*** *selon la définition dans [23] : étant donnée une croyance $U \rightarrow V$, on cherche une règle $X \rightarrow Y$ telle que :*

$$un(X,Y) \equiv \begin{cases} freq(Y \cup V) = 0 \wedge \\ freq(X \cup U) \geq minfr_1 \wedge \\ freq(X \cup U \cup Y) \geq minfr_2 \wedge \\ freq(X \cup U \cup Y)/freq(X \cup U) \geq minconf \wedge \\ (freq(X \cup U \cup V) < maxfr \vee \\ freq(X \cup U \cup V)/freq(X \cup U) < maxconf) \end{cases}$$

où $minfr_1$, $minfr_2$ et $maxfr$ sont des seuils de fréquence et $minconf$ et $maxconf$ sont des seuils de confiance.

Chapitre 2

CSP : notions de base

Ce chapitre introductif présente les notions de base relatives aux CSPs, dont nous aurons besoin dans ce mémoire. Tout d'abord, nous présentons les notions de CSP, de cohérence et les mécanismes de filtrage associés, ainsi que différentes méthodes de recherche.

2.1 Le formalisme des CSPs

2.1.1 Variables et domaines

Soit $\mathcal{X}$ un ensemble fini de n variables $\mathcal{X}$=$\{X_1, \ldots, X_n\}$. À chaque variable X_i est associée un domaine, noté D_{X_i} , représentant l'ensemble fini des valeurs pouvant être prises par cette variable. Le domaine d'une variable peut être numérique {1,3,4} ou symbolique {Matin,Soir,Repos}. On désigne l'ensemble des domaines par D = $\{D_{X_1}, \ldots, D_{X_n}\}$.

Définition 19 (affectation d'une variable)**.** *On appelle* ***affectation d'une variable*** X_i*, le fait d'attribuer à* X_i *une valeur de son domaine.*

L'affectation de la variable X_i à la valeur v_j est notée $(X_i{=}v_j)$. Une affectation complète A=$\{$ $(X_1{=}v_2),\ldots, (X_n{=}v_n)\}$ est une affectation de toutes les variables de X. Lorsque qu'au moins une variable n'est pas affectée, on parlera d'affectation partielle (notée $\mathcal{A}_p$).

2.1.2 Contraintes

Soit $\mathcal{C}$ un ensemble contenant e contraintes. Chaque contrainte $c \in \mathcal{C}$ porte sur un ensemble de variables noté X_c. Cet ensemble de variables est appelé *portée*[1] de la contrainte c.

Une contrainte est une relation qui impose des conditions sur les valeurs qui peuvent être affectées aux variables de sa portée. Ces restrictions peuvent être exprimées de manière symbolique (par exemple $X_1 < X_2$), dans ce cas on parle de *contrainte en intention*. Elles peuvent aussi être exprimées sous la forme d'un ensemble de *tuples autorisés*, c'est-à-dire l'ensemble des affectations des variables satisfaisant la contrainte. On parle dans ce cas de *contrainte en extension* ou *de contrainte table*.

1. La portée d'une contrainte c est aussi appelée *scope* de c.

Définition 20 (contrainte satisfaite)**.** *Une contrainte* c *est dite* ***satisfaite*** *ssi les variables* X_c *sont complètement instanciées et forment un tuple vérifiant* c*.*

Soit les deux variables X_1 et X_2 de domaines $D_{X_1} = D_{X_2} = \{1\,;\,2\}$, la contrainte $X_1 \neq X_2$ est satisfaite si X_1 est affectée à la valeur 1 et X_2 à 2 (ou inversement).

Un tuple autorisé est une affectation de toutes les variables de la portée d'une contrainte c satisfaisant celle-ci. Par opposition, un *tuple interdit* pour une contrainte c correspond à une affectation complète des variables de X_c ne satisfaisant pas c.

Les contraintes portant sur une variable sont dites unaires. Les variables portant sur deux variables, trois variables et n variables sont respectivement appelées *binaires*, *ternaires* et *n-aires*.

2.1.3 Problème de satisfaction de contraintes

Une instance du problème de satisfaction de contraintes [2] (CSP) [21, 19] est définie par un triplet ($\mathcal{X}$,$\mathcal{D}$,$\mathcal{C}$), avec $\mathcal{X}$ l'ensemble des variables, $\mathcal{D}$ l'ensemble des domaines associés aux variables de $\mathcal{X}$ et $\mathcal{C}$ l'ensemble des contraintes. Un CSP dont la portée maximale des contraintes est 2 est dit binaire.

▪ **Exemple 5.** *Soit le réseau de contrainte P= ($\mathcal{X}$,$\mathcal{D}$,$\mathcal{C}$), avec* $\mathcal{X}$=$\{X_1, X_2, X_3, X_4\}$, $\mathcal{D}$=$\{D_{X_1}, D_{X_2}, D_{X_3}, D_{X_4}\}$, D_{X_1}=D_{X_2}=$\{1, 2, 3\}$, D_{X_3}=$\{2, 3\}$, D_{X_4}=$\{2, 3, 4\}$ *et* $\mathcal{C}$=$\{X_1 = X_2, X_1 = X_3, X_2 = X_3, X_1 + X_2 \leq 3, X_3 < X_4\}$.

Définition 21 (solution d'un CSP)**.** *Une* ***solution d'un CSP*** *P est une instanciation complète qui satisfait toutes les contraintes de P.*

Le CSP P (Exemple 1) possède deux solutions :
— { (X_1=1), (X_2=2), (X_3=3), (X_4=4)} ;
— { (X_1=2), (X_2=1), (X_3=3), (X_4=4)}.

2.2 Méthodes de cohérence

Dans le formalisme des CSPs, les contraintes restreignent l'espace de recherche afin de définir les solutions. En s'appuyant sur ces contraintes, plusieurs mécanismes de cohérence ont été proposés afin d'interdire ou de *filtrer* des valeurs (ou des combinaisons de valeurs), et afin de propager les conséquences des retraits. Nous donnons ci-dessous une description rapide de quelques mécanismes de filtrage.

2.2.1 Cohérence de nœud

La nœud cohérence est le mécanisme de filtrage le plus simple permettant de vérifier que les contraintes unaires sont satisfaisables.

Définition 22 (Cohérence de nœud)**.** *Une variable* X_i *est dite nœud cohérent si et seulement si chaque valeur de son domaine satisfait toutes les contraintes unaires portant sur* X_i*. Un CSP est nœud cohérent si et seulement si toutes ses variables le sont et qu'aucun domaine n'est vide.*

2. Par abus de langage, nous utiliserons le terme CSP pour désigner une instance du problème de satisfaction de contraintes.

2.2.2 Arc cohérence

L'arc cohérence permet de filtrer les valeurs des domaines en tenant compte des contraintes binaires. Pour cela, elle se base sur la notion de support définie ci-dessous.

Définition 23 (Support d'une valeur d'une variable pour une contrainte binaire)**.** *Soient c_{ij} une contrainte binaire portant sur X_i et X_j, a une valeur de D_i et b une valeur de D_j. b est dite support de $(X_i = a)$ pour c_{ij} si et seulement si le tuple (a, b) est autorisé. L'ensemble des supports de $(X_i = a)$ pour c_{ij} est noté $support(c_{ij}, X_i, a)$.*

Définition 24 (Viabilité d'une valeur)**.** *Une valeur a d'une variable X_i est* viable *si et seulement si pour toute contrainte binaire c_{ij}, il existe au moins un support de a dans le domaine de X_j.*

Définition 25 (Arc cohérence d'un CSP, [42, 13])**.** *Un CSP est arc cohérent si et seulement si toutes les valeurs des variables sont viables et qu'aucun domaine n'est vide.*

La cohérence d'arc est une cohérence *locale* à chaque contrainte. Un CSP peut donc tre arc cohérent et n'avoir aucune solution. La figure 2.1 représente le graphe de micro-structures d'un tel CSP.

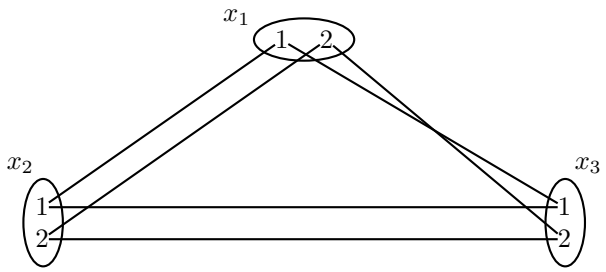

FIGURE 2.1 – Exemple de CSP arc cohérent insatisfaisable.

Afin de rendre un problème arc cohérent, de nombreux mécanismes ont été proposés. Parmi ceux-ci, nous pouvons citer `AC-4` qui fût la première méthode optimale dans le pire cas proposée par [20]. Sa complexité temporelle pour un graphe binaire est $O(e \times d^2)$, avec e le nombre de contraintes et d la taille du plus grand domaine. Cependant, celle-ci est peu utilisée en pratique en raison d'une complexité spatiale non négligeable ($O(e \times d^2)$) et d'une phase d'initialisation en $O(e \times d^2)$ coûteuse (i.e. qui a la même complexité que l'algorithme lui-même). Afin de pallier ces problèmes, plusieurs améliorations ont été proposées :

- `AC-6` ([6]) permet de diminuer, grâce à un calcul paresseux des supports, la complexité spatiale à $O(e \times d)$ et la complexité temporelle en moyenne (la complexité temporelle dans le pire cas étant préservée).
- `AC-7` ([7]) permet pour des contraintes bidirectionnelles de diminuer le nombre de vérifications de cohérence lors de la résolution.
- `AC-2000` ([8]) basé sur le principe de `AC-3` [19] permet de vérifier la cohérence par perte ou présence de support suivant le cas le plus avantageux.
- `AC-2001` ([8]) utilise une structure de données additionnelle permettant de stocker le dernier support trouvé. Lors du prochain test de cohérence, on testera en premier si

ce support est toujours présent dans le domaine et si tel est le cas, aucune recherche de support n'est effectuée. Cet algorithme a une complexité temporelle dans le pire cas optimale $O(e \times d^2)$ et une complexité spatiale de $O(e \times d)$.

2.3 Recherche de solution

Dans le cadre des problèmes de satisfaction de contraintes, on peut distinguer trois types de problèmes fondamentaux :

— l'existence d'une solution ;
— la recherche d'une solution ;
— le comptage et/ou la recherche de toutes les solutions d'un problème.

Le problème de l'existence d'une solution est un problème NP-Complet. Les deux problèmes de la recherche d'une solution et du comptage de toutes les solutions sont des problèmes de difficulté plus élevée.

Dans la suite de cette mémoire, nous nous intéresserons essentiellement au problème de la recherche d'une solution. Nous utiliserons l'exemple suivant : **Exemple 2 :** Le but est de trouver une solution au problème du carré latin de dimension 3 de la figure 2.2.

1		
	1	

FIGURE 2.2 – Carré latin de dimension 3.

Arbre de recherche

i) Structure d'un arbre de recherche

L'espace de recherche peut être représenté par un arbre : à chaque noeud de l'arbre (ou point de choix) correspond une variable et à chaque branche une décision. Suivant la nature du point de choix et le nombre de branches sortant de celui-ci, nous parlons de branchement :

— d-aire : chaque branche issue d'un même noeud représente une des différentes affectations possibles de la variable ;
— binaire : à chaque point de choix, deux branches sont associées. Sur la première branche, le choix d'affecter la valeur v_j à la variable X_i est fait $(X_i = v_j)$, et sur la seconde le choix de ne pas affecter cette variable à cette valeur $(X_i \neq v_j)$;
— par séparation de domaine : chaque décision réduit le domaine de la variable (sans forcément l'affecter). Par exemple, pour D_{X_1}={1, ..., 10}, la contrainte $X_1 > 4$ peut être ajoutée sur une branche et $X_1 \leq 4$ sur une autre.

Chaque feuille de profondeur n représente une instanciation complète résultant des instanciations faites le long de la branche menant à cette feuille. Si cette instanciation complète satisfait toutes les contraintes du problème alors il s'agit d'une solution.

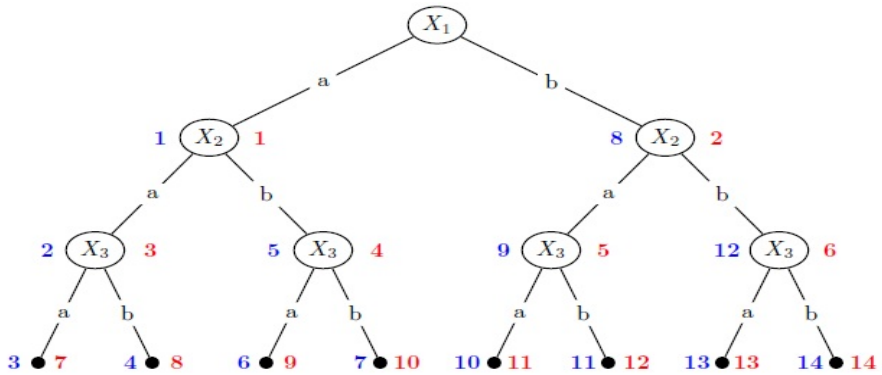

FIGURE 2.3 – Les valeurs à gauche des noeuds représentent l'ordre de parcours des noeuds selon la stratégie en profondeur d'abord et celles de droite en largeur d'abord.

ii) Stratégies d'exploration

Il existe plusieurs stratégies pour explorer un arbre de recherche :

— *le parcours en profondeur d'abord* (depth first) : cette stratégie favorise la descente dans l'arbre de recherche en explorant en premier les noeuds les plus à gauche dans l'arbre de recherche.

— *le parcours en largeur d'abord* (breadth first) : l'arbre est exploré par niveau de la gauche vers la droite.

— *le parcours en utilisant le meilleur d'abord* (best first) : cette stratégie utilise la gestion d'une frontière entre les noeuds explorés et ceux non explorés (cette frontière peut être représentée par une file à priorités). Au moment de choisir un noeud à explorer, le noeud de la frontière qui est le plus prometteur selon un critère d'évaluation est sélectionné. Une fois un noeud traité, tous ses fils sont ajoutés à la frontière.

La figure 2.3 présente sur un exemple l'ordre d'exploration des noeuds d'un arbre de recherche selon les stratégies en profondeur d'abord, et en largeur d'abord.

Dans la suite de cette section, nous utiliserons la stratégie de parcours en profondeur d'abord [3].

Sur l'exemple du carré latin de dimension 3 de la figure 2.2, plus de 1400 branches doivent être explorées avant d'atteindre une première solution, soit environ 65% de l'espace de recherche.

3. Nous utiliserons l'ordre lexicographique pour ordonner les variables et les valeurs.

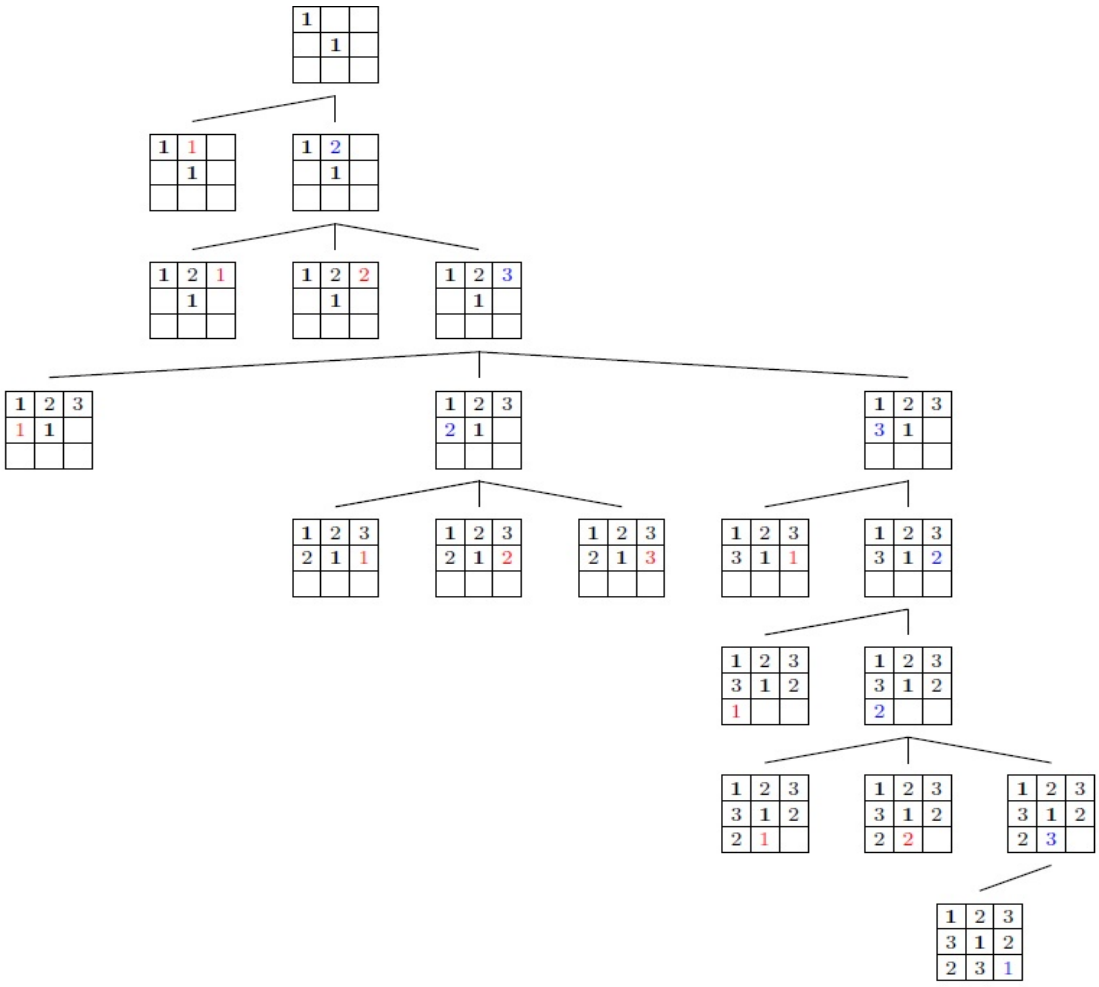

FIGURE 2.4 – Arbre de recherche.

Chapitre 3

Relaxation de contraintes

De nombreux problèmes réels sont par nature sur-contraints (ils ne possèdent aucune solution). Cela est souvent dû à un ensemble d'exigences trop fortes des utilisateurs de l'application. Ainsi, dans les problèmes d'emplois du temps, les souhaits ou préférences exprimés par les élèves, les enseignants et l'administration sont souvent incompatibles. Différents modèles ou cadres pour la relaxation des CSPs ont été proposés : les CSPs hiérarchiques [11], les Partial CSPs [14], les CSPs valués [38], les Semi-ring CSPs [10], la relaxation disjonctive [25], ...

Le cadre que nous avons retenu pour modéliser la relaxation dans l'extraction de motifs sous contraintes est le modèle disjonctif [25]. Dans ce chapitre, nous présentons successivement la problématique, le modèle disjonctif et les raisons qui ont motivé notre choix.

3.1 Problématique

Relaxer une contrainte (quelconque) c'est l'autoriser à ne pas être nécessairement satisfaite contre un coût. Lorsque l'on cherche à modéliser un problème sur-contraint, on distingue deux catégories de contraintes.

1. Les *contraintes d'intégrité* (ou dures) doivent être impérativement satisfaites. Les contraintes d'intégrité reflètent généralement des obligations d'ordre physique.
   Exemples : un enseignant ne peut faire deux cours en même temps, une salle ne peut accueillir deux cours en même temps, ...
2. Les *contraintes de préférence* (ou souples) expriment des souhaits formulés sous forme de propriétés que l'on aimerait voir vérifiées par une solution. Ces contraintes de préférence correspondent à des souhaits qu'il faudrait respecter pour obtenir une solution de bonne qualité.
   Exemples : certains enseignants souhaitent ne pas faire plus de six heures de cours par jour, les étudiants souhaitent que les cours commencent après 9 heures, ...

Le but de la relaxation n'est plus de satisfaire toutes les contraintes (cela n'est pas toujours possible), mais de les satisfaire *au mieux*, i.e. satisfaire toutes les contraintes dures et minimiser une agrégation des coûts des contraintes de préférence insatisfaites. Ainsi, la relaxation se modélise sous forme d'un problème d'optimisation (COP).

3.2 Relaxation disjonctive d'un réseau de contraintes

Thierry Petit et al. proposent dans [25, 26] de transformer la résolution d'un problème surcontraint (COP) en un problème de satisfaction afin de bénéficier du savoir faire sur les CSPs. Pour cela, les coûts sont intégrés directement au problème via l'ajout d'un ensemble de variables.

Définition 26 (variable de coût associée à une contrainte [25]). *Soit c_i une contrainte, la variable de coût z_i (associée à c_i) est une variable à valeurs numériques positives ou nulle telle que :*
- *si c_i est satisfaite alors $z_i = 0$,*
- *si c_i est insatisfaite alors $z_i > 0$.*

La valeur de z_i quantifie la violation et dépend de la sémantique de violation retenue pour la contrainte c_i.

Définition 27 (sémantique de violation). *μ est une sémantique de violation pour la contrainte $c(X_1, ..., X_n)$ ssi μ est une fonction de $D_{X_1} \times ... \times D_{X_n}$ vers $\mathbb{R}^+$ telle que : $\forall \mathcal{A} \in D_{X_1} \times ... \times D_{X_n}$, $\mu(\mathcal{A}) = 0$ ssi $c(X_1, ..., X_n)$ est satisfaite.*

▪ **Exemple 6.** *Soit la contrainte binaire c_i définie par $X_1 = X_2$, avec pour domaines $D_{X_1} = D_{X_2} = \{1, 2, 3\}$. Si l'on choisit comme sémantique de violation la distance entre les deux variables, alors $z_i = |X_1 - X_2|$. Les coûts de violation de différentes instanciations sont présentés dans les tableaux suivants :*

X_1	X_2	z_i
1	1	0
1	2	1
1	3	2

X_1	X_2	z_i
2	1	1
2	2	0
2	3	1

X_1	X_2	z_i
3	1	2
3	2	1
3	3	0

Coûts de violation pour l'exemple 6

La version relaxée de chaque contrainte est formulée sous forme d'une disjonction : soit la contrainte est vérifiée et le coût est nul, soit la contrainte est insatisfaite et le coût est précisé.

Définition 28 (relaxation disjonctive d'une contrainte [25]). *Soit $R=(\mathcal{X}, \mathcal{D}, \mathcal{C})$ un réseau de contraintes, $c_i \in \mathcal{C}$ une contrainte, $\bar{c}_i$ sa négation et z_i sa variable de coût. La relaxation disjonctive de c_i est la contrainte c_{disj_i} définie par :*

$$c_{disj_i} = [[c_i \wedge [z_i = 0]] \vee [\bar{c}_i \wedge [z_i > 0]]$$

▪ **Exemple 7.** *La relaxation disjonctive de la contrainte c_i de l'exemple 6 est la suivante :*

$$c_{disj_i} = [[X_1 = X_2 \wedge z_i = 0] \vee [X_1 \neq X_2 \wedge z_i = |X_1 - X_2|]$$

Soit C_s l'ensemble des contraintes souples et C_h l'ensemble des contraintes d'intégrité. A chaque contrainte $c_i \in C_s$, on associe z_i qui est sa variable de coût. Soit z la variable représentant la violation totale (cumul des violations), alors $z = \sum_{c_i \in C_s} z_i$. Soit λ la quantité maximale de violation que l'on s'autorise. On doit avoir : $z \leq \lambda$, i.e. $\sum_{c_i \in C_s} z_i \leq \lambda$. Le domaine de z est réduit à l'intervalle $[0..\lambda]$.

Définition 29 (relaxation disjonctive d'un réseau de contraintes [25]). *Soit $R=(\mathcal{X},\mathcal{D},\mathcal{C})$ un réseau de contraintes, la relaxation disjonctive de R est le réseau de contraintes $R'=(\mathcal{X}',\mathcal{D}',\mathcal{C}')$ dérivé de R tel que :*

- $\mathcal{X}' = \mathcal{X} \cup \mathcal{X}_Z \cup \{z\}$
- $\mathcal{D}' = \mathcal{D} \cup \mathcal{D}_Z \cup \{[0..\lambda]\}$
- $\mathcal{C}' = \mathcal{C}_h \cup \mathcal{C}_{disj} \cup \{z = \oplus_{i=1}^{|C_{disj}|} z_i\}$

avec

- $\mathcal{X}_\mathcal{Z} = \{z_1, \ldots, z_{|C_s|}\} \cup \{z\}$, *l'ensemble des variables de coût,*
- $\mathcal{D}_\mathcal{Z} = \{D_{z_1}, \ldots, D_{z_{|C_s|}}\} \cup \{[0..\lambda]\}$, *l'ensemble des domaines des variables de coût,*
- C_{disj} *l'ensemble des contraintes de préférence exprimées sous forme disjonctive,*
- C_h *l'ensemble des contraintes d'intégrité.*

▪ **Exemple 8.** *Soit le CSP $P=(\mathcal{X},\mathcal{D},\mathcal{C})$ tel que :*

- $\mathcal{X} = \{X_1, X_2, X_3\}$,
- $\mathcal{D} = \{D_{X_1} = \{2,3\}, D_{X_2} = \{1,2\}, D_{X_3} = \{1,2,3\}\}$,
- $\mathcal{C} = \{c_1 = [X_1 > X_2], c_2 = [X_1 = X_3], c_3 = [X_3 > X_2]\}$

On suppose que :

- c_1 est une contrainte d'intégrité.
- c_2 et c_3 sont des contraintes de préférence.
- La sémantique de violation de c_2 est la distance entre les valeurs de ses deux variables.
- La sémantique de violation de c_3 est le carré de l'écart entre les valeurs de ses deux variables.

La relaxation disjonctive de P est le réseau de contrainte $P'=(\mathcal{X}',\mathcal{D}',\mathcal{C}')$ tel que :

- $\mathcal{X}' = \mathcal{X} \cup \{z_1, z_2, z\}$
- $\mathcal{D}' = \mathcal{D} \cup \{D_{z_1}, D_{z_2}, D_z\}$
- $\mathcal{C}' = \{X_1 > X_2,$
  $[X_1 = X_3 \wedge z_1 = 0] \vee [X_1 \neq X_3 \wedge z_1 = |X_1 - X_3|],$
  $[X_3 > X_2 \wedge z_2 = 0] \vee [X_3 \leq X_2 \wedge z_2 = (X_3 - X_2)^2]$
  $z = z_1 + z_2\}$

Remarque : Cette forme générale est difficilement exploitable en raison des disjonctions apparues. Dans certains cas, on peut simplifier cette écriture de manière drastique, comme nous le montrerons dans la section 4.3.

3.3 Motivations de notre choix

Le premier apport du modèle disjonctif est que les variables de coût permettent de contrôler la violation de manière simple et explicite : équilibrage de la violation, limitation de la violation d'une contrainte, ou encore ajout de contraintes sur les variables de coût. L'ajout de contraintes portant sur les variables initiales et les variables de coût (méta contraintes) permettra d'exprimer ce contrôle de la violation.

Un autre point fort de ces méta contraintes est qu'elles peuvent, grâce au filtrage, réduire les domaines des variables de coût et ainsi provoquer la réduction des domaines des variables de la contrainte initiale.

Le formalisme des WCSPs est une solution alternative. Mais, dans cette phase exploratoire, on ne les a pas choisis car les solveurs de WCSPs ne savent prendre en compte que les contraintes binaires ou ternaires (Pour une comparaison plus fine entre les WCSPs et le modèle disjonctif, se reporter à la thèse de Jean-Philippe Métivier section 3.3).

Le modèle proposé par Thierry PETIT et al. possède à la fois la possibilité d'exprimer des mesures de violation fines et d'exprimer facilement des règles de contrôle de la violation (notamment grâce aux variables de coût). De plus le modèle disjonctif ne possède pas de limitation sur les contraintes utilisables (ni sur le nombre de variables, ni sur la nature de la contrainte). Les bonnes propriétés de ce modèle nous ont conduit à le retenir pour la relaxation de contraintes pour l'extraction de motifs.

Chapitre 4

Relaxation de contraintes de seuil

4.1 Problématique et état de l'art

Soient X un motif local, $\{m_i\}$ un ensemble de mesures, $\{\alpha_i\}$ un ensemble des seuils (correspondant aux mesures $\{m_i\}$) et une requête exprimée sous la forme d'une conjonction de contraintes de la forme $m_i(X) \geq \alpha_i$ (ou bien $\leq$).
On se pose 2 questions :

- Si un motif X satisfait presque toutes les contraintes de la requête, pourquoi le rejeter ?
- Comment sont fixées les valeurs des seuils qui doivent vérifier les mesures ?

D'où l'idée d'accepter, comme solutions, les motifs qui ne satisfont pas la requête, mais qui sont "proches". Pour cela, nous souhaitons approximer l'ensemble des solutions de la contrainte originale (notée c) par une collection de motifs plus large correspondant à l'ensemble de solutions d'une contrainte moins restrictive $c^{'}$ déduite de c.

L'idée sous-jacente est d'obtenir une relaxation vérifiant le cadre disjonctif. Plus précisément, **étant donné un langage $\mathcal{L}$, une base de données $\mathcal{T}$ et une contrainte c, nous souhaitons obtenir automatiquement une relaxation de** c. Une telle approche est une méthode d'optimisation qui préserve la complétude [5] puisque l'élagage issu de la relaxation ne rejette pas de motifs satisfaisant c.

Dans la littérature, la relaxation est aussi utilisée pour découvrir des motifs plus inattendus [2] ou introduire de la souplesse afin d'éviter une sélection trop binaire (effet "crisp" [9]). D'autres travaux utilisent la relaxation pour obtenir des contraintes "relâchées" ayant des propriétés de monotonie dans le but de réutiliser les algorithmes usuels de filtrage. Ainsi, les contraintes basées sur des expressions régulières sont relaxées en des contraintes anti-monotones pour extraire des séquences [15]. Dans le domaine des ensembles, une large collection de formules booléennes de contraintes monotones [39] et de contraintes d'agrégats [43] peuvent être relaxées. Dans [39], étant donnée une contrainte c, les auteurs proposent de générer automatiquement une *relaxation monotone* et *anti-monotone* de la contrainte c. Comparé à toutes ces méthodes, **nous proposons une approche générique en relaxant n'importe quelle type des contrainte. Par ailleurs, notre méthode est complètement automatisable**.

4.2 Sémantiques de violation pour les contraintes de seuil

Comme nous avons a vu dans la définition 27, on peut définir des sémantiques de violation permettant de quantifier la violation sur des mesures utilisées pour l'extraction de motifs.

Soit $\mathcal{I}$ un ensemble de n items et $\mathcal{T}$ un ensemble de m transactions sur $\mathcal{I}$. Soit X un motif local et α un seuil.

Afin de décrire notre approche, nous l'illustrons par 2 exemples avant de présenter le cas général :

4.2.1 Exemple introductif : $c_1 \equiv freq(X) \geq \alpha$

Une **première sémantique de violation** μ_1 pour la contrainte c_1 est d'associer, à chaque motif X, l'écart (absolu) de sa fréquence au seuil α :

$$\mu_1(X) = \begin{cases} 0 & si\ freq(X) \geq \alpha \\ \alpha - freq(X) & sinon \end{cases}$$

Une **seconde sémantique de violation** [1] μ_2 pour la contrainte c_1 est d'associer, à chaque motif X, l'écart (relatif) de sa fréquence au seuil α :

$$\mu_2(X) = \begin{cases} 0 & si\ freq(X) \geq \alpha \\ \frac{\alpha - freq(X)}{\alpha} & sinon \end{cases}$$

Une **troisième sémantique de violation** μ_3 pour la contrainte c est définie comme suit : si la fréquence d'un motif X est jugée trop loin du seuil, alors on considère que la contrainte est insatisfaite.

On s'autorise à relaxer dans une proportion de $(r \times \alpha)$, où r est un pourcentage ($r \in [0..1]$). Il y a désormais 3 cas :

1. $freq(X) \geq \alpha$ alors la contrainte est satisfaite et $\mu_3(X) = 0$.
2. $(1 - r) \times \alpha \leq freq(X) \leq \alpha$ alors la contrainte est relaxée et $\mu_3(X) = (\alpha - freq(X))/\alpha$,
3. $freq(X) < (1 - r) \times \alpha$ alors la contrainte est insatisfaite.

On a alors : $\forall X \in \mathcal{L}_{\mathcal{I}}, \mu_3(X) \in [0..r]$, avec :

$$\mu_3(X) = \begin{cases} 0 & si\ freq(X) \geq \alpha \\ \frac{\alpha - freq(X)}{\alpha} & si\ (1 - r) \times \alpha \leq freq(X) \leq \alpha \\ \infty & sinon \end{cases}$$

1. Si l'on veut sommer des violations issues de plusieurs contraintes de fréquence, il vaut mieux "normer" à l'aide de la violation maximale (ici α).

4.2.2 2nd exemple : $c_2 \equiv freq(X) \leq \alpha$

Une **première sémantique de violation** μ_1 pour la contrainte c est d'associer à chaque motif X, l'écart (absolu) de sa fréquence au seuil α :

$$\mu_1(X) = \begin{cases} 0 & si\ freq(X) \leq \alpha \\ freq(X) - \alpha & sinon \end{cases}$$

Une **seconde sémantique de violation** μ_2 pour la contrainte c_2 est d'associer à chaque motif X, l'écart (relatif) de sa fréquence au seuil α ($m - \alpha$ étant la violation maximale pour la mesure $freq(X)$) :

$$\mu_2(X) = \begin{cases} 0 & si\ freq(X) \leq \alpha \\ \frac{freq(X)-\alpha}{m-\alpha} & sinon \end{cases}$$

Une **troisième sémantique de violation** μ_3 pour la contrainte c_2 est définie comme suit : si la fréquence d'un motif X est jugée trop loin du seuil, alors on considère que la contrainte est insatisfaite.

On s'autorise à relaxer dans une proportion de $(r \times \alpha)$, où r est un pourcentage ($r \in [0..1]$). Il y a désormais 3 cas :

1. $freq(X) \leq \alpha$ alors la contrainte est satisfaite et $\mu_3(X) = 0$.
2. $\alpha \leq freq(X) \leq (1 + r) \times \alpha$ alors la contrainte est relaxée et $\mu_3(X) = \frac{freq(X)-\alpha}{\alpha}$,
3. $(1 + r) \times \alpha < freq(X)$ alors la contrainte est insatisfaite.

$$\mu_3(X) = \begin{cases} 0 & si\ freq(X) \leq \alpha \\ \frac{freq(X)-\alpha}{\alpha} & si\ \alpha \leq freq(X) \leq (1 + r) \times \alpha \\ \infty & sinon \end{cases}$$

4.2.3 Cas d'une mesure quelconque $m(X)$

On peut généraliser les trois sémantiques de violation précédemment décrites à une mesure quelconque m et un seuil α pour cette mesure.

— Une **première sémantique de violation** μ_1 pour la contrainte c est d'associer à chaque motif X, l'écart (absolu) de sa fréquence au seuil α :

$$\rightarrow c \equiv m(X) \geq \alpha \qquad \rightarrow \mu_1(X) = \begin{cases} 0 & si\ m(X) \geq \alpha \\ \alpha - m(X) & sinon \end{cases}$$

$$\rightarrow c \equiv m(X) \leq \alpha \qquad \rightarrow \mu_1(X) = \begin{cases} 0 & si\ m(X) \leq \alpha \\ m(X) - \alpha & sinon \end{cases}$$

— Une **seconde sémantique de violation** μ_2 [2] pour la contrainte c est d'associer à chaque motif X, l'écart (relatif) de sa fréquence au seuil α :

$$\rightarrow c \equiv m(X) \geq \alpha \qquad \rightarrow \mu_2(X) = \begin{cases} 0 & si\ m(X) \geq \alpha \\ \frac{\alpha - m(X)}{\alpha} & sinon \end{cases}$$

$$\rightarrow c \equiv m(X) \leq \alpha \qquad \rightarrow \mu_2(X) = \begin{cases} 0 & si\ m(X) \leq \alpha \\ \frac{m(X) - \alpha}{max_m - \alpha} & sinon \end{cases}$$

— Une **troisième sémantique de violation** μ_3 pour la contrainte c est d'associer :

$$\rightarrow c \equiv m(X) \geq \alpha \qquad \rightarrow \mu_3(X) = \begin{cases} 0 & si\ m(X) \geq \alpha \\ \frac{\alpha - m(X)}{\alpha} & si\ (1-r) \times \alpha \leq m(X) \leq \alpha \\ \infty & sinon \end{cases}$$

$$\rightarrow c \equiv m(X) \leq \alpha \qquad \rightarrow \mu_3(X) = \begin{cases} 0 & si\ m(X) \leq \alpha \\ \frac{m(X) - \alpha}{\alpha} & si\ \alpha \leq m(X) \leq (1+r) \times \alpha \\ \infty & sinon \end{cases}$$

4.3 Transformation : Cas des motifs locaux

4.3.1 Exemple introductif : $freq(X) \geq \alpha$ (cf Section 4.2.1)

a) Cas de la sémantique μ_1

Soit $c_i \equiv (freq(X) \geq \alpha)$ et sa sémantique de violation μ_1. Sa relaxation disjonctive est :

$$c'_i \equiv [(freq(X) \geq \alpha) \wedge z_i = 0] \vee [(freq(X) < \alpha) \wedge z_i = \alpha - freq(X)]$$

Que l'on peut simplifier en :

$$c'_i \equiv [z_i = max(0, \alpha - freq(X))]$$

b) Cas de la sémantique μ_2

Soit $c_i \equiv (freq(X) \geq \alpha)$ et sa sémantique de violation μ_2. Sa relaxation disjonctive est :

$$c'_i \equiv [(freq(X) \geq \alpha) \wedge z_i = 0] \vee [(freq(X) < \alpha) \wedge z_i = \frac{\alpha - freq(X)}{\alpha}]$$

Que l'on peut simplifier en :

$$c'_i \equiv [z_i = max(0, \frac{\alpha - freq(X)}{\alpha})]$$

2. max_m est la valeur maximale pour la mesure m (pour la $frequence$ $max_m = m$, pour la $taille$, $max_m = n$, etc)

c) Cas de la sémantique μ_3
Soit $c_i \equiv (freq(X) \geq \alpha)$ et sa sémantique de violation μ_3. Sa transformation est :

$$c'_i \equiv [freq(X) \geq (1-r) \times \alpha] \wedge [(freq(X) \geq \alpha) \wedge z_i = 0] \vee [(freq(X) < \alpha) \wedge z_i = \frac{\alpha - freq(X)}{\alpha}]$$

Que l'on peut simplifier en :

$$c'_i \equiv [freq(X) \geq (1-r) \times \alpha] \wedge [z_i = max(0, \frac{\alpha - freq(X)}{\alpha})]$$

4.3.2 2nd exemple : $freq(X) \leq \alpha$ (cf Section 4.2.2)

a) Cas de la sémantique μ_2
Soit $c_i \equiv (freq(X) \leq \alpha)$ et sa sémantique de violation μ_2. Sa relaxation disjonctive est :

$$c'_i \equiv [(freq(X) \leq \alpha) \wedge z_i = 0] \vee [(freq(X) > \alpha) \wedge z_i = \frac{freq(X) - \alpha}{m - \alpha}]$$

Que l'on peut simplifier en :

$$c'_i \equiv [z_i = max(0, \frac{freq(X) - \alpha}{m - \alpha})]$$

b) Cas de la sémantique μ_3
Soit $c_i \equiv (freq(X) \leq \alpha)$ et sa sémantique de violation μ_3. Sa transformation est :

$$c'_i \equiv [freq(X) \leq (1+r) \times \alpha] \wedge [(freq(X) \leq \alpha) \wedge z_i = 0] \vee [(freq(X) > \alpha) \wedge z_i = \frac{freq(X) - \alpha}{\alpha}]$$

Que l'on peut simplifier en :

$$c'_i \equiv [freq(X) \leq (1+r) \times \alpha] \wedge [z_i = max(0, \frac{freq(X) - \alpha}{\alpha})]$$

4.3.3 Cas d'une mesure quelconque $m(X)$ (cf Section 4.2.3)

a) Cas de la sémantique μ_2
— Soit $c_i \equiv (m(X) \geq \alpha)$ Sa relaxation disjonctive est :

$$c'_i \equiv [(m(X) \geq \alpha) \wedge z_i = 0] \vee [(m(X) < \alpha) \wedge z_i = \frac{\alpha - m(X)}{\alpha}]$$

Que l'on peut simplifier en :

$$c'_i \equiv [z_i = max(0, \frac{\alpha - m(X)}{\alpha})]$$

— Soit $c_i \equiv (m(X) \leq \alpha)$ Sa relaxation disjonctive est :

$$c'_i \equiv [(m(X) \leq \alpha) \wedge z_i = 0] \vee [(m(X) > \alpha) \wedge z_i = \frac{m(X) - \alpha}{max_m - \alpha}]$$

Que l'on peut simplifier en :

$$c'_i \equiv [z_i = max(0, \frac{m(X) - \alpha}{max_m - \alpha})]$$

b) Cas de la sémantique μ_3

— Soit $c_i \equiv (m(X) \geq \alpha)$ Sa transformation est :

$$c'_i \equiv [m(X) \geq (1-r) \times \alpha] \wedge [(m(X) \geq \alpha) \wedge z_i = 0] \vee [(m(X) < \alpha) \wedge z_i = \frac{\alpha - m(X)}{\alpha}]$$

Que l'on peut simplifier en :

$$c'_i \equiv [m(X) \geq (1-r) \times \alpha] \wedge [z_i = max(0, \frac{\alpha - m(X)}{\alpha})]$$

— Soit $c_i \equiv (m(X) \leq \alpha)$ Sa transformation est :

$$c'_i \equiv [m(X) \leq (1+r) \times \alpha] \wedge [(m(X) \leq \alpha) \wedge z_i = 0] \vee [(m(X) > \alpha) \wedge z_i = \frac{m(X) - \alpha}{\alpha}]$$

Que l'on peut simplifier en :

$$c'_i \equiv [m(X) \leq (1+r) \times \alpha] \wedge [z_i = max(0, \frac{m(X) - \alpha}{\alpha})]$$

4.3.4 Transformation d'une requête

Pour chaque contrainte c_i de la requête, faire :

— si c_i est une contrainte dure alors elle reste telle quelle.

— si c_i est une contrainte souple alors elle est remplacée par sa transformée :

1. pour la sémantique μ_2, la transformée est une contrainte de coût.
2. pour la sémantique μ_3, la transformée est la conjonction d'une contrainte dure et d'une contrainte de coût.

Ainsi, toute requête contenant une ou plusieurs contraintes souple peut être transformée en une requête **équivalente** ne contenant que des contraintes dures et pouvant donc être résolue par un solveur de contraintes.

Enfin, pour quantifier la violation globale, on définit une nouvelle variable de coût Z représentant le cumul des violations, puis on ajoute la contrainte $Z \leq \lambda$, avec $Z = \sum_{c_i} z_i$ et $D_Z = [0..\lambda]$, où z_i est la variable de coût associé à chaque contrainte souple c_i et λ la quantité maximale de violation autorisée.

Motivations pour l'extraction de motif n-aires

Le cadre de l'extraction de motifs sous contraintes est un paradigme puissant pour découvrir de nouvelles connaissances très précieuses et exprimer l'intérêt de l'utilisateur. Le paradigme de la programmation par contraintes apporte des techniques utiles pour exprimer un tel intérêt.

Si l'extraction de motifs locaux est désormais un domaine bien maîtrisé, y compris des approches génériques, ces méthodes ne prennent pas en compte l'intérêt d'un motif par rapport aux autres motifs qui sont extraits : les motifs utiles sont perdus parmi les informations triviales, bruitées et redondantes. En pratique, beaucoup de motifs qui sont attendus par l'analyste de données nécessitent d'examiner simultanément plusieurs motifs pour combiner les informations fragmentées par les motifs locaux.

Exploiter la localité des motifs pour extraire des motifs globaux tels que classificateurs ou clustering est devenu, ces dernières années, un domaine de recherche émergent et prometteur. Dans la suite, de tels motifs seront appelés *motifs n-aires*, et une requête impliquant des motifs n-aires sera appelée *requête n-aire*.

Comme nous l'avons vu dans la section 1.2, peu de travaux concernant l'extraction de motifs n-aires ont été menés et les méthodes développées sont toutes *ad hoc*C'est pourquoi nous nous intéressons dans ce stage au cas des *motifs n-aires*.

4.4 Transformation : Cas des motifs n-aires

Pour ce cas on utilise le même principe que pour les motifs locaux, c'est-à-dire que pour relaxer une requête n-aire (conjonction de contraintes n-aires) on fait la même transformation : Pour chaque contrainte c_i de la requête, faire :

— si c_i est *dure* alors elle reste telle quelle.

— si c_i est *souple* alors elle est transformée.

Enfin, comme pour le cas des motifs locaux, on ajoute la contrainte $Z = \sum_{c_i} z_i \leq \lambda$.

4.4.1 Exemple : Règles d'exception

Comme nous l'avons vu en section 1.2.1, les règles d'exception sont composées de 2 parties : une règle générale ($X \backslash Y \rightarrow I$) et une règle déviationnelle ($X \rightarrow \neg I$). Nous avons choisi de relaxer la règle générale car on sait mieux fixer les seuils de la règle déviationnelle, alors qu'il est plus difficile de fixer les seuils de la règle générale. Nous rappelons ci-dessous la définition de la règle d'exception :

$$exception(X,Y,I) \equiv \begin{cases} freq((X \setminus Y) \sqcup I) \geq minfr \wedge \\ freq(X \setminus Y) - freq((X \setminus Y) \sqcup I) \leq \delta_1 \\ \wedge \\ freq(X \sqcup \neg I) \leq maxfr \wedge \\ freq(X) - freq(X \sqcup \neg I) \leq \delta_2 \end{cases}$$

$minfr$ et $maxfr$: seuils de fréquence
δ_1 et δ_2 : seuils de confiance

D'où les contraintes qui définissent la règle générale :

$$règle\ générale(X,Y,I) \equiv \begin{cases} freq((X \setminus Y) \sqcup I) \geq minfr \wedge \\ freq(X \setminus Y) - freq((X \setminus Y) \sqcup I) \leq \delta_1 \end{cases}$$

Pour relaxer ces deux contraintes, on utilise la sémantique de violation μ_3 :

1. Pour la contrainte $freq((X \setminus Y) \sqcup I) \geq minfr$ (la règle générale est fréquente), on lui associe la variable de coût z_1 et on applique la transformation suivante (cf. Section 4.3.1c) :

$$z_1 = \max(0, \tfrac{minfr - freq((X \backslash Y) \sqcup I)}{minfr}) \wedge freq((X \setminus Y) \sqcup I) \geq (1-r) \times minfr$$

2. Pour la contrainte $freq(X \setminus Y) - freq((X \setminus Y) \sqcup I) \leq \delta_1$ (la règle générale est de forte confiance), on associe la variable de coût z_2 et on utilise la transformation (cf Section 4.3.2c) :

$$z_2 = \max(0, \frac{(freq(X \setminus Y) - freq((X \setminus Y) \sqcup I) - \delta_1}{\delta_1}) \wedge freq(X \setminus Y) - freq((X \setminus Y) \sqcup I) \leq (1 + r) \times \delta_1$$

Enfin, on définit une variable de coût Z représentant le cumul des violations, puis on ajoute la contrainte $Z \leq \lambda$ au modèle CSP, avec $Z = z_1 + z_2$ et $D_Z = [0..\lambda]$.

4.4.2 Exemple : Règles inattendues

Les règles inattendues sont définies par :

$$un(X, Y) \equiv \begin{cases} freq(Y \cup V) = 0 \wedge \\ freq(X \cup U) \geq minfr_1 \wedge \\ freq(X \cup U \cup Y) \geq minfr_2 \wedge \\ freq(X \cup U \cup Y)/freq(X \cup U) \geq minconf \wedge \\ (freq(X \cup U \cup V) < maxfr \vee \\ freq(X \cup U \cup V)/freq(X \cup U) < maxconf) \end{cases}$$

Nous avons choisi de relaxer les trois premières contraintes :

(i) $freq(Y \cup V) = 0$;

(ii) $freq(X \cup U) \geq minfr_1$;

(iii) $freq(X \cup U \cup Y) \geq minfr_2$;

La relaxation de la contrainte (i) est motivée par le fait qu'il s'agit d'une contrainte dure qui est très difficile à satisfaire dans la pratique. Pour les deux autres contraintes, cela est dû, d'une part, à la difficulté de fixer chacun des deux seuils individuellement pour d'obtenir les contraintes correspondantes et d'autre part, à la difficulté de donner un seuil approprié à $minfr_1$ par rapport au seuil $minfr_2$.

La relaxation de la contrainte (i) est définie comme suit :

— On définit la variable de coût z_1 et on applique la relaxation disjonctive suivante :

$$c' \equiv [freq(Y \cup V) = 0 \wedge z_1 = 0] \vee [freq(Y \cup V) > 0 \wedge z_1 = freq(Y \cup V)]$$

Que l'on peut simplifier en : $c' \equiv [z_1 = freq(Y \cup V)]$

Pour les deux autres contraintes, on utilise la sémantique de violation μ_1 :

— Pour la contrainte $freq(X \cup U) \geq minfr_1$ (XU est un motif fréquent), on définit la variable de coût z_2 et on obtient la transformée ci-dessous (cf Section 4.3.1a) :

$$z_2 = \max(0, minfr_1 - freq(X \cup U))$$

— Pour la contrainte $freq(X \cup U \cup Y) \geq minfr_2$ ($XU \rightarrow Y$ est une règle fréquente), on définit la variable de coût z_3 et on obtient la transformée suivante (cf Section 4.3.1a) :

$$z_3 = \max(0, minfr_2 - freq(X \cup U \cup Y))$$

Finalement, on définit la variable de coût global Z et on ajoute la contrainte $Z \leq \lambda$ au CSP, avec $Z = z_1 + z_2 + z_3$ et $D_Z = [0..\lambda]$, où λ est le seuil de violation maximale.

Chapitre 5

Mise en œuvre

Ce chapitre décrit l'implantation de notre modèle de relaxation pour le cas des motifs n-aires. Pour cela, nous avons utilisé l'implantation de l'extracteur de motifs FIM_CP [1] qui est une approche PPC pour l'extraction de motifs locaux [34]. Cette approche traite dans un cadre unifié un large ensemble de motifs locaux et de contraintes telles que la fréquence, la fermeture, la maximalité, les contraintes monotones ou anti-monotones et des variantes de ces contraintes.

Cet extracteur a été étendu par Mehdi Khiari pour l'extraction de motifs n-aires et cette approche a été mise en œuvre en Gecode [16].

5.1 Formulation générale

La recherche de motifs ensemblistes peut se modéliser à l'aide d'un CSP $\mathcal{P} = (\mathcal{X}, \mathcal{D}, \mathcal{C})$:

- $\mathcal{X} = \{X_1, \ldots, X_k\}$. Chaque variable X_i représente un motif ensembliste inconnu.
- $\mathcal{D} = \{D_{X_1}, \ldots, D_{X_k}\}$. Le domaine initial de chaque variable X_i est $[\{\}, \ldots, \mathcal{I}]$
- $\mathcal{C} = \mathcal{C}_{ens} \cup \mathcal{C}_{num}$:
  - $\mathcal{C}_{ens}$ est une conjonction de contraintes ensemblistes formulées à l'aide d'opérateurs ensemblistes ($\cup$,$\cap$,$\setminus$,$\in$,$\notin$,...)
  - $\mathcal{C}_{num}$ est une conjonction de contraintes numériques telles que : ($<$,$\leq$,$\neq$,$=$,...)

Méthode de transformation

Pour faire la relaxation de contraintes de seuil pour l'extraction de motifs n-aires, nous avons défini une méthode de transformation comme suit :

1. Modélisation de la règle choisie sous forme d'un CSP.
2. Obtention du CSP associé à la relaxation dans le cadre disjonctif.
3. Implantation.

1. `http://www.cs.kuleuven.be/~dtai/CP4IM/fim\_cp.php`

5.2 Étape 1 : Modélisation de la règle choisie sous forme d'un CSP

5.2.1 Règles d'exception

D'après la définition 17, nous avons :

$$exception(X,Y) \equiv \begin{cases} freq((X \setminus Y) \sqcup I) \geq minfr \wedge \\ freq(X \setminus Y) - freq((X \setminus Y) \sqcup I)) \leq \delta_1 \\ \wedge \\ freq(X \sqcup \neg I) \leq maxfr \wedge \\ freq(X) - freq(X \sqcup \neg I) \leq \delta_2 \end{cases}$$

Que l'on peut formuler avec des contraintes primitives comme suit :

Contrainte	Formulation
$freq((X \setminus Y) \sqcup I) \geq minfr$	$freq(X_2) \geq minfr$ $\wedge\ I \in X_2$ $\wedge\ X_1 \subsetneq X_3$
$freq(X \setminus Y) - freq((X \setminus Y) \sqcup I) \leq \delta_1$	$freq(X_1) - freq(X_2) \leq \delta_1$ $\wedge\ X_2 = X_1 \sqcup I$
$freq(X \sqcup \neg I) \leq maxfr$	$freq(X_4) \leq maxfr$ $\wedge\ \neg I \in X_4$
$freq(X) - freq(X \sqcup \neg I) \leq \delta_2$	$freq(X_3) - freq(X_4) \leq \delta_2$ $\wedge\ X_4 = X_3 \sqcup \neg I$

TABLE 5.1 – Formulation des contraintes

La table 5.1 décrit l'ensemble des contraintes primitives modélisant les règles d'exception, que l'on peut décrire par le CSP $\mathcal{P}_1 = (\mathcal{X}_1, \mathcal{D}_1, \mathcal{C}_1)$ ci-dessous :

a) $\mathcal{X}_1 = \{X_1, X_2, X_3, X_4\}$ est l'ensemble des variables qui représentent les motifs recherchés :
   — $X_1 : X \setminus Y$, et $X_2 : (X \setminus Y) \sqcup I$ (règle générale) et
   — $X_3 : X$, et $X_4 : X \sqcup \neg I$ (règle déviationnelle).

b) $\mathcal{D}_1 = \{D_{X_1}, D_{X_2}, D_{X_3}, D_{X_4}\}$ est l'ensemble des domaines où $D_{X_1} = D_{X_2} = D_{X_3} = D_{X_4} = \{\emptyset, \ldots, \mathcal{I}\}$.

c) $\mathcal{C}_1 = \mathcal{C}^1_{ens} \cup \mathcal{C}^1_{num}$ est l'ensemble des contraintes :
   — Les contraintes ensemblistes : $\mathcal{C}^1_{ens} = \{(I \in X_2), (X_2 = X_1 \sqcup I), (\neg I \in X_4), (X_4 = X_3 \sqcup \neg I), (X_1 \subsetneq X_3)\}$.
   — Les contraintes numériques : $\mathcal{C}^1_{num} = \{(freq(X_2) \geq minfr), (freq(X_1) - freq(X_2) \leq \delta_1), (freq(X_4) \leq maxfr), (freq(X_3) - freq(X_4) \leq \delta_2)\}$.

5.2.2 Règle inattendue

D'après la définition 18, nous avons :

$$un(X,Y) \equiv \begin{cases} freq(Y \cup V) = 0 \wedge \\ freq(X \cup U) \geq minfr_1 \wedge \\ freq(X \cup U \cup Y) \geq minfr_2 \wedge \\ freq(X \cup U \cup Y)/freq(X \cup U) \geq minconf \wedge \\ (freq(X \cup U \cup V) < maxfr \vee \\ freq(X \cup U \cup V)/freq(X \cup U) < maxconf) \end{cases}$$

Que l'on peut formuler avec des contraintes primitives comme suit :

Contrainte	Formulation
$freq(Y \cup V) = 0$	$freq(X_2) = 0$ $X_2 = X_1 \sqcup X_7$ $X_1 = Y$ $X_7 = V$
$freq(X \cup U) \geq minfr_1$	$freq(X_3) \geq minfr_1$ $X_3 = X_0 \sqcup X_6$ $X_0 = X$ $X_6 = U$
$freq(X \cup U \cup Y) \geq minfr_2$	$freq(X_4) \geq minfr_2$ $X_4 = X_3 \sqcup X_1$
$\frac{freq(X \cup U \cup Y)}{freq(X \cup U)} \geq minconf$	$\frac{freq(X_4)}{freq(X_3)} \geq minconf$
$freq(X \cup U \cup V) < maxfr$ $\vee$ $\frac{freq(X \cup U \cup V)}{freq(X \cup U)} < maxconf$	$freq(X_5) < maxfr$ $X_5 = X_3 \sqcup X_7$ $\vee$ $\frac{freq(X_8)}{freq(X_6)} < maxconf$ $X_8 = X_6 \sqcup X_7$

TABLE 5.2 – Formulation des contraintes pour les règles inattendues

La table 5.2 décrit l'ensemble des contraintes primitives modélisant les règles inattendues que l'on peut décrire par le CSP $\mathcal{P}_2 = (\mathcal{X}_2, \mathcal{D}_2, \mathcal{C}_2)$ ci-dessous :

a) $\mathcal{X}_2 = \{X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7, X_8\}$ est l'ensemble des variables où :
- $X_0 : X$,
- $X_1 : Y$,
- $X_2 : Y \cup V$,
- $X_3 : X \cup U$,
- $X_4 : X \cup U \cup Y$,
- $X_5 : X \cup V$,

— $X_6 : U$,
— $X_7 : V$,
— $X_8 : U \cup V$.

b) $\mathcal{D}_2 = \{D_{X_0}, D_{X_1}, D_{X_2}, D_{X_3}, D_{X_4}, D_{X_5}, D_{X_6}, D_{X_7}, D_{X_8}\}$ est l'ensemble des domaines où $D_{X_0} = D_{X_1} = D_{X_2} = D_{X_3} = D_{X_4} = D_{X_5} = D_{X_6} = D_{X_7} = D_{X_8} = \{\emptyset, \ldots, \mathcal{I}\}$.

c) $\mathcal{C}_2 = \mathcal{C}^2_{ens} \cup \mathcal{C}^2_{num}$ est l'ensemble des contraintes :
— Les contraintes ensemblistes : $\mathcal{C}^2_{ens} = \{(X_2 = X_1 \sqcup X_7), (X_1 = Y), (X_7 = V), (X_3 = X_0 \sqcup X_6), (X_0 = X), (X_6 = U), (X_4 = X_3 \sqcup X_1), (X_5 = X_3 \sqcup X_7), (X_8 = X_6 \sqcup X_7)\}$.
— Les contraintes numériques : $\mathcal{C}^2_{num} = \{(freq(X_2) = 0), (freq(X_3) \geq minfr_1), (freq(X_4) \geq minfr_2), (\frac{freq(X_4)}{freq(X_3)} \geq minconf), (freq(X_5) < maxfr), (\frac{freq(X_8)}{freq(X_6)} < maxconf)\}$.

5.3 Étape 2 : Obtention du CSP associé à la relaxation dans le cadre disjonctif

5.3.1 Règle d'exception

Maintenant, on définit le CSP $\mathcal{P}'_1 = (\mathcal{X}'_1, \mathcal{D}'_1, \mathcal{C}'_1)$ associé à $\mathcal{P}_1$, où :

a) $\mathcal{X}'_1 = \mathcal{X}_1 \cup \{z_1, z_2, Z\}$ est l'ensemble des variables :
— Les variables de coût $\{z_1,z_2\}$ quantifient la violation pour les contraintes souples et la variable de coût globale Z sert à quantifier la violation globale.

b) $\mathcal{D}'_1 = \mathcal{D}_1 \cup \{D_{z_1}, D_{z_2}, D_Z\}$ est l'ensemble des domaines :
— Comme on utilise la sémantique de violation μ_3 (écart relatif restreint), alors les valeurs des variables z_1 et z_2 sont normalisées, donc les domaines des variables de coût $D_{z_1} = D_{z_2} = \{0,\ldots, 1\}$ et $D_Z = \{0,\ldots,\lambda\}$.

c) $\mathcal{C}'_1 = \mathcal{C}^1_{ens} \cup \mathcal{C}^{1'}_{num} \cup \{Z = \sum z_i\} \cup \{Z \leq \lambda\}$ est l'ensemble des contraintes :
— Contraintes numériques : $\mathcal{C}^{1'}_{num} = \mathcal{C}^1_{hard} \cup \mathcal{C}^{1'}_{soft}$ où :
— $\mathcal{C}^1_{hard} = \{(freq(X_4) \leq maxfr), (freq(X_3) - freq(X_4) \leq \delta_2)\}$ est l'ensemble des contraintes dures
— $\mathcal{C}^1_{soft} = \{(freq(X_2) \geq minfr), (freq(X_1) - freq(X_2) \leq \delta_1)\}$ est l'ensemble des contraintes souples dont la relaxation est : $\mathcal{C}^{1'}_{soft} = \{(z_1 = max(0, \frac{minfr - freq(X_2)}{minfr})), (z_2 = max(0, \frac{freq(X_1) - freq(X_2)}{\delta_1}$

5.3.2 Règle inattendue

Maintenant, on définit le CSP $\mathcal{P}'_2 = (\mathcal{X}'_2, \mathcal{D}'_2, \mathcal{C}'_2)$ associé à $\mathcal{P}_2$, où :

a) $\mathcal{X}'_2 = \mathcal{X}_2 \cup \{z_1, z_2, z_3, Z\}$ désigne l'ensemble des variables :
— Les variables de coût $\{z_1,z_2,z_3\}$ quantifient la violation pour les contraintes souples et la variable de coût globale Z sert à quantifier la violation globale.

b) $\mathcal{D}'_2 = \mathcal{D}_2 \cup \{D_{z_1}, D_{z_2}, D_{z_3}, D_Z\}$ désigne l'ensemble des domaines :
— Comme on utilise la sémantique de violation μ_1 (écart absolu) et les variables souples portent sur la mesure de $frequence$, alors les variables z_1, z_2 et z_3 ont, au plus, la valeur m, donc les domaines des variables de coût $D_{z_1} = D_{z_2} = D_{z_3} = \{0,\ldots, m\}$ et $D_Z = \{0,\ldots,\lambda\}$.

c) $\mathcal{C}'_2 = \mathcal{C}^2_{ens} \cup \mathcal{C}^{2'}_{num} \cup \{Z = \sum z_i\} \cup \{Z \leq \lambda\}$ désigne l'ensemble de contraintes :

— Contraintes numériques : $\mathcal{C}^{2'}_{num} = \mathcal{C}^{2}_{hard} \cup \mathcal{C}^{2'}_{soft}$ où :
  — $\mathcal{C}^{2}_{hard} = \{(\frac{freq(X_4)}{freq(X_3)} \geq minconf), (freq(X_5) < maxfr), (\frac{freq(X_8)}{freq(X_6)} < maxconf)\}$ est l'ensemble des contraintes dures.
  — $\mathcal{C}^{2}_{soft} = \{(freq(X_2) = 0), (freq(X_3) \geq minfr_1), (freq(X_4) \geq minfr_2)\}$ est l'ensemble des contraintes souples dont la relaxation est : $\mathcal{C}^{2'}_{soft} = \{(z_1 = freq(X_2)), (z_2 = max(0, minfr_1 - freq(X_3)), (z_3 = max(0, minfr_2 - freq(X_4))\}$.

5.4 Étape 3 : Implantation

5.4.1 Le cadre CP4IM - Khiari

a) **Modélisation des k motifs n-aires recherchés :**
Soit d la matrice 0/1 de dimension (m,n) telle que $\forall t \in \mathcal{T}$,$\forall i \in \mathcal{I}$, $(d_{t,i} = 1)$ ssi $(i \in t)$. Soient X_1,X_2,...,X_k les k motifs n-aires recherchés.
Tout d'abord, chaque motif n-aire inconnu X_j est modélisé par n variables de décision $\{X_{1,j}, X_{2,j}, \ldots, X_{n,j}\}$ (de domaine {0,1}) telles que $(X_{i,j} = 1)$ ssi l'item i appartient au motif X_j :

$$\forall i \in \mathcal{I}, (X_{i,j} = 1) \Longleftrightarrow (i \in X_j) \tag{5.1}$$

Puis, m variables de décision $\{T_{1,j}$,$T_{2,j}$,...,$T_{m,j}\}$ (de domaine {0,1}) sont associées à chaque motif n-aire inconnu X_j telles que $(T_{t,j} = 1)$ ssi $(X_j \subseteq t)$:

$$\forall t \in \mathcal{T}, (T_{t,j} = 1) \Longleftrightarrow (X_j \subset t) \tag{5.2}$$

La relation entre le motif recherché X_j et $\mathcal{T}$ est établie via des contraintes réifiées imposant que, pour chaque transaction t, $(T_{t,j} = 1)$ ssi $(X_j \subseteq t)$, ce qui se reformule en :

$$\forall j \in [1 \ldots k], \forall t \in \mathcal{T}, (T_{t,j} = 1) \Longleftrightarrow \sum_{i \in \mathcal{I}} X_{i,j} \times (1 - d_{t,i}) = 0 \tag{5.3}$$

Ainsi, les mesures $freq$ et $taille$ se formulent comme suit :

$$freq(X_j) = \sum_{t \in \mathcal{T}} T_{t,j} \quad \text{et} \quad taille(X_j) = \sum_{i \in \mathcal{I}} X_{i,j} \tag{5.4}$$

▪ **Exemple 9.** *Considérons la base de données $\mathcal{T} = \{t_1, t_2, \ldots, t_9\}$ et $\mathcal{I} = [1..6]$ (cf. Table 1.2), où les items A, B, C, D, c_1, c_2, sont numérotés de 1 à 6. Alors nous avons :*

$\forall j \in [1..k]$,
$(T_{1,j} = 1) \Leftrightarrow (X_{3,j} = 0 \wedge X_{4,j} = 0 \wedge X_{6,j} = 0)$
$(T_{2,j} = 1) \Leftrightarrow (X_{3,j} = 0 \wedge X_{4,j} = 0 \wedge X_{6,j} = 0)$
$(T_{3,j} = 1) \Leftrightarrow (X_{1,j} = 0 \wedge X_{2,j} = 0 \wedge X_{4,j} = 0 \wedge X_{6,j} = 0)$
$(T_{4,j} = 1) \Leftrightarrow (X_{1,j} = 0 \wedge X_{2,j} = 0 \wedge X_{4,j} = 0 \wedge X_{6,j} = 0)$
$(T_{5,j} = 1) \Leftrightarrow (X_{1,j} = 0 \wedge X_{2,j} = 0 \wedge X_{4,j} = 0 \wedge X_{6,j} = 0)$
$(T_{6,j} = 1) \Leftrightarrow (X_{5,j} = 0)$
$(T_{7,j} = 1) \Leftrightarrow (X_{1,j} = 0 \wedge X_{2,j} = 0 \wedge X_{5,j} = 0)$
$(T_{8,j} = 1) \Leftrightarrow (X_{1,j} = 0 \wedge X_{2,j} = 0 \wedge X_{4,j} = 0 \wedge X_{5,j} = 0)$
$(T_{9,j} = 1) \Leftrightarrow (X_{1,j} = 0 \wedge X_{2,j} = 0 \wedge X_{3,j} = 0 \wedge X_{5,j} = 0)$

b) **Contraintes numériques et ensemblistes :**

Considérons un opérateur $op \in \{<, \leq, >, \geq, =, \neq\}$; les contraintes numériques se reformulent comme suit :

— $freq(X_p)\ op\ \alpha \rightarrow \sum_{t \in \mathcal{T}} T_{t,p}\ op\ \alpha$

— $taille(X_p)\ op\ \alpha \rightarrow \sum_{i \in \mathcal{I}} X_{i,p}\ op\ \alpha$

Certaines contraintes ensemblistes (telles que égalité, inclusion, appartenance,...) se reformulent directement à l'aide de contraintes linéaires :

— $X_p = X_q \rightarrow \forall i \in \mathcal{I}, X_{i,p} = X_{i,q}$

— $X_p \subseteq X_q \rightarrow \forall i \in \mathcal{I}, X_{i,p} \leq X_{i,q}$

— $i_0 \in X_p \rightarrow X_{i_0,p} = 1$

Les autres contraintes ensemblistes (telles que intersection, union différence,...) se reformulent aisément à l'aide de contraintes booléennes en utilisant la fonction de conversion ($b :: \{0,1\} \rightarrow \{False, True\}$) et les opérateurs booléens usuels :

— $X_p \cap X_q = X_r \rightarrow \forall i \in \mathcal{I}, b(X_{i,r}) = b(X_{i,p}) \wedge b(X_{i,q})$

— $X_p \cup X_q = X_r \rightarrow \forall i \in \mathcal{I}, b(X_{i,r}) = b(X_{i,p}) \vee b(X_{i,q})$

— $X_p \setminus X_q = X_r \rightarrow \forall i \in \mathcal{I}, b(X_{i,r}) = b(X_{i,p}) \wedge \neg b(X_{i,q})$

Enfin, l'ensemble des contraintes, qu'elles soient réifiées, numériques ou ensemblistes, est traité par *Gecode*.

5.4.2 Exemple : Règle d'exception

Comme décrit en Section 5.3.1, le CSP $\mathcal{P}'_1 = (\mathcal{X}'_1, \mathcal{D}'_1, \mathcal{C}'_1)$ est constitué par l'ensmble des contraintes numériques $\mathcal{C}^{1'}_{num} = \mathcal{C}^1_{hard} \cup \mathcal{C}^{1'}_{soft}$, avec :

— $\mathcal{C}^1_{hard} = \{(freq(X_4) \leq maxfr), (freq(X_3) - freq(X_4) \leq \delta_2)\}$ est l'ensemble des contraintes dures,

— $\mathcal{C}^1_{soft} = \{(freq(X_2) \geq minfr), (freq(X_1) - freq(X_2) \leq \delta_1)\}$ est l'ensemble des contraintes souples dont la relaxation est : $\mathcal{C}^{1'}_{soft} = \{(z_1 = max(0, \frac{minfr - freq(X_2)}{minfr})), (z_2 = max(0, \frac{freq(X_1) - freq(X_2) - \delta_1}{\delta_1}$

D'après la section 5.4.1b, toutes les contraintes numériques se reformulent comme suit :

$$\forall i \in [1..k], freq(X_i)\ op\ \alpha_i \rightarrow \sum_{t \in \mathcal{T}} T_{t,i}\ op\ \alpha_i \tag{5.5}$$

Donc :

— $freq(X_4) \leq maxfr \rightarrow \sum_{t \in \mathcal{T}} T_{t,4} \leq maxfr$,

— $freq(X_3) - freq(X_4) \leq \delta_2 \rightarrow \sum_{t \in \mathcal{T}} T_{t,3} - \sum_{t \in \mathcal{T}} T_{t,4} \leq \delta_2$,

— $z_1 = max(0, \frac{minfr - freq(X_2)}{minfr} \rightarrow z_1 = max(0, \frac{minfr - \sum_{t \in \mathcal{T}} T_{t,2}}{minfr})$,

— $z_2 = max(0, \frac{freq(X_1) - freq(X_2) - \delta_1}{\delta_1} \rightarrow z_2 = max(0, \frac{\sum_{t \in \mathcal{T}} T_{t,1} - \sum_{t \in \mathcal{T}} T_{t,1} - \delta_1}{\delta_1})$.

De la même façon, on peut modéliser et implanter toutes les contraintes de seuils pour l'extraction de motifs (locaux ou n-aires), en suivant cette démarche.

Chapitre 6

Expérimentations

Ce chapitre a pour but de montrer la faisabilité et les apports pratiques de l'introduction de la relaxation de contraintes sur les approches PPC déjà développés pour l'extraction de motifs (locaux ou n-aires).

6.1 Protocole expérimental

Différents expérimentations ont été menées sur plusieurs jeux de données de l'*UCI repository*[1] utilisés par le projet CP4IM[2], ainsi que sur un jeu de données réel (noté **Meningitis**) issu de de l'Hôpital Central de Grenoble. **Meningitis** recense les pathologies des enfants atteints d'une méningite virale ou bactérienne. La table 6.1 résume les différentes caractéristiques des jeux de données utilisés.

Les expérimentations ont été menées sur deux exemples de contraintes n-aires : les règles d'exception et les règles inattendues. La machine utilisée est un PC 2.13 GHz Intel (R) Core (TM) i3 processor avec 4GB RAM, sous Ubuntu Linux.

Jeu de données	#trans	#items	densité
Mushroom	8142	119	0.18
SoyBean	630	50	0.32
Tic-tac-toe	958	27	0.33
Meningitis	329	84	0.25

TABLE 6.1 – Description des jeux de données

Comme la résolution effectuée par le solveur de contraintes est correcte et complète, notre approche permet d'extraire l'ensemble correct et complet des motifs satisfaisant la transformation en contraintes souples de n'importe quelle contrainte n-aire (cf Section 4.3).

1. `http://www.ics.uci.edu/~mlearn/MLRepository`
2. `http://dtai.cs.kuleuven.be/CP4IM/`

6.2 Règles d'exception

Pour introduire la relaxation dans l'extraction de règles d'exception, nous avons retenu la sémantique de violation μ_3 (c'est-à-dire écart relatif restreint). Comme cette sémantique de violation nécessite l'introduction d'un paramètre $\boldsymbol{r}$ appelé *écart de violation*, nous avons distingué deux cas :

- même écart de violation pour toutes les contraintes à relaxer (un seul paramètre $\boldsymbol{r}$ donné par l'utilisateur pour toutes les contraintes) ;
- différents écarts de violation, un pour chaque contrainte à relaxer (un paramètre r_i donné par l'utilisateur pour chaque contrainte).

6.2.1 Même écart de violation pour toutes les contraintes

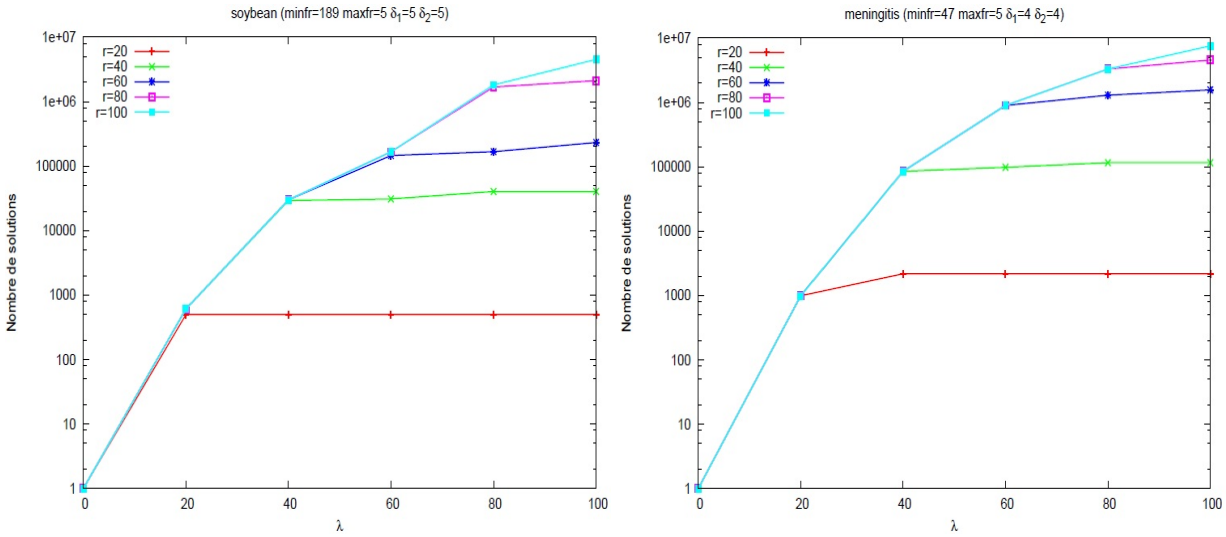

FIGURE 6.1 – Évolution du nombre de règles d'exception

La figure 6.1 décrit l'évolution du nombre de paires de règles d'exception en fonction des paramètres λ (seuil de violation maximale) et r (écart de violation) pour les jeux de données **Soybean** et **Meningitis**. Nous avons aussi testé d'autres jeux de données. Mais, comme les résultats obtenus sont similaires, ils ne sont pas indiqués ici.

Pour les besoins de nos expérimentations, nous avons initialisé le paramètre λ à r (i.e., λ=r) pour 2 raisons principales :

- pour minimiser les nombres de paramètres donnés par l'utilisateur. En effet, en plus de seuils $minfr$, $maxfr$, δ_1 et δ_2, l'utilisateur doit également définir le paramètre r.
- pour imposer à chaque contrainte d'être violée au plus par un écart r et que la violation globale ne peut dépasser ce même écart.

Notons que pour $r = 0$, nous nous ramenons au cas dur (i.e., aucune contrainte n'est relaxée). Dans ce cas, comme on peut le constater sur la figure 6.1, il n'existe pas solutions satisfaisant les contraintes (avec des seuils durs) de la règle d'exception. En revanche, en relaxant les seuils de fréquence et de confiance de la règle générale, nous arrivons à trouver des solutions "proches".

Comme attendu, plus la valeur de r est grande, plus le nombre de règles d'exception augmente. En effet, quand r augmente, les seuils de la fréquence et de la confiance de la règle générale décroisent et il y a donc un plus grand nombre de règles générales.

6.2.2 Différents écarts de violation pour les contraintes

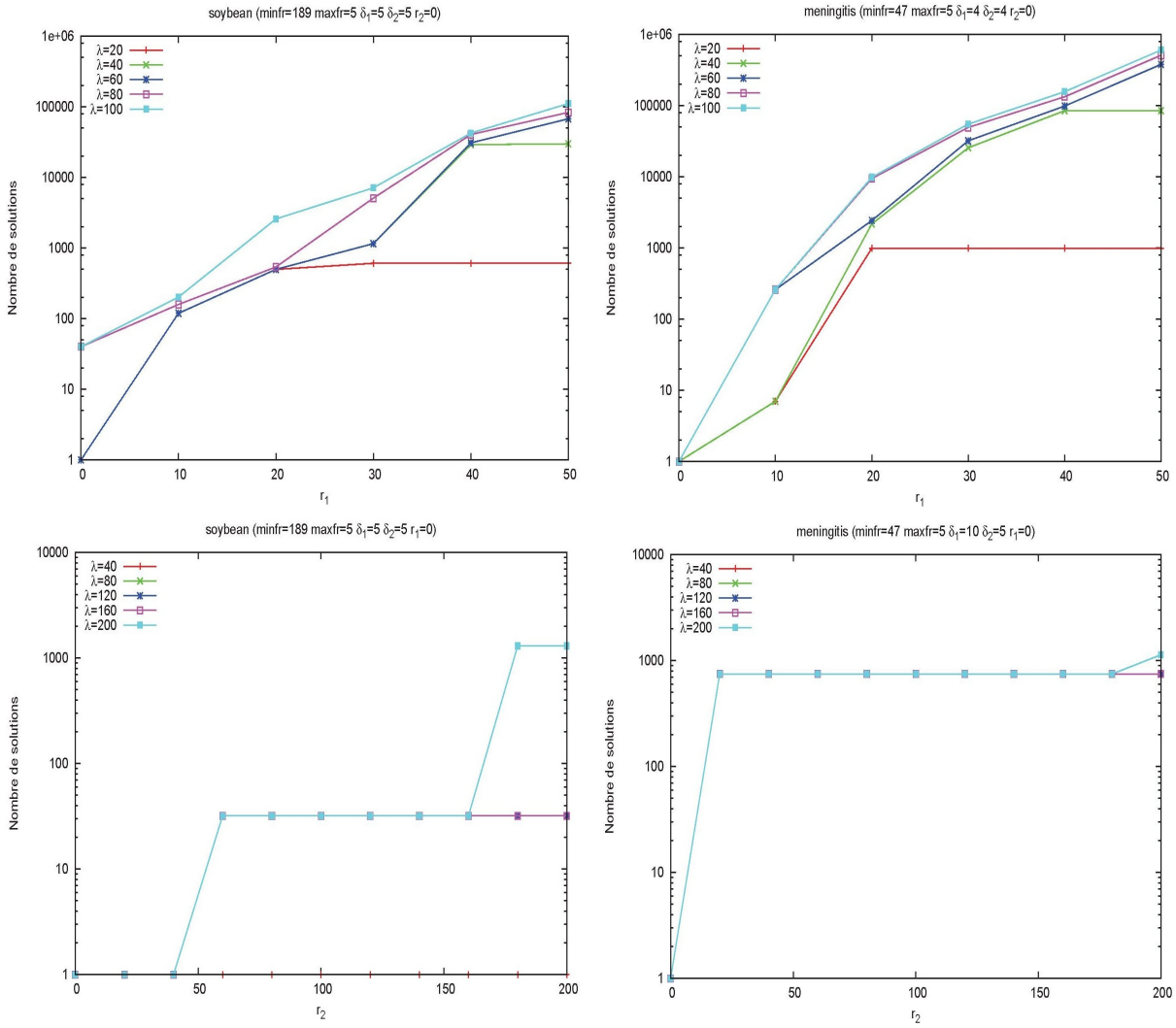

FIGURE 6.2 – Évolution du nombre de règles d'exception

Pour ce second cas, nous avons réalisé deux séries d'expérimentations. Tout d'abord, dans un premier temps, nous autorisons uniquement la relaxation du seuil de fréquence de la règle générale (i.e., $r_2 = 0$). Puis, dans un second temps, nous relaxons uniquement le seuil de confiance (i.e., $r_1 = 0$).

La figure 6.2 décrit l'évolution du nombre de paires de règles d'exception en fonction des valeurs des paramètres λ et r_i $(i = 1, 2)$ pour les jeux de données **Soybean** et **Meningitis**. Une

fois de plus, nous retrouvons le même comportement que précédemment, à savoir, plus r_i est grand, plus le nombre de règles d'exception augmente. Par ailleurs, nous constatons que plus la valeur de λ est grand, plus la courbe représentant cette valeur est dominante.

Sur le jeu de données **Soybean**, on peut remarquer que l'écart de violation associé au seuil de confiance (paramètre r_2) est beaucoup plus important dans le cas d'un seuil de fréquence dur (i.e., $r_1 = 0$). Ce qui n'est pas le cas pour un seuil de confiance dur (i.e., $r_2 = 0$), où de petits écarts de violation du paramètre r_1 sont suffisants pour trouver des premières solutions.

Mise en valeur de motifs demandés par les utilisateurs

Les règles d'exception sont un cas particulier des règles rare. D'après le travail effectué par M. Khiari [17] même s'il existe quelques travaux permettant d'extraire les règles rares [41], il est impossible de distinguer les règles d'exception à partir de l'ensemble de règles rares. C'est une limitation forte car la plupart des règles rares sont peu fiables, d'où l'intérêt des règles d'exception et de leur extraction. Savoir rechercher directement les règles d'exception permet de réduire de manière drastique le nombre de motifs obtenus.

Mais pour certaines requêtes on n'obtient pas de solution, car l'utilisateur ne connaît pas *a priori* les bonnes valeurs pour les seuils, d'où l'intérêt de trouver des solutions "*proches*" pour pouvoir tirer de conclusions de ces solutions, à la différence du cadre dure pour lequel on n'aurait aucune solution.

6.3 Règles inattendues

Pour introduire la relaxation dans l'extraction de règles inattendues, nous avons choisi d'utiliser la sémantique de violation μ_1 (écart absolu), car les trois contraintes relaxées sont toutes des contraintes de fréquence, donc il n'est pas nécessaire de normaliser l'écart pour pouvoir agréger les valeurs des variables de coût z_i.

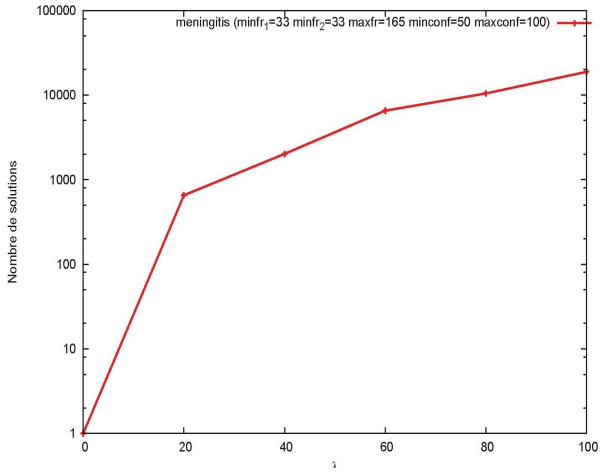

FIGURE 6.3 – Évolution du nombre de règles inattendues

Les expérimentations ont été menées sur le jeu de données réel **Meningitis** (cf. la table 6.1). La figure 6.3 décrit l'évolution du nombre de paires de règles inattendues en fonction du seuil λ (seuil de violation maximale). Pour $\lambda = 0$ (aucune contrainte n'est relaxée), aucune solution n'a pu être trouvée sur ce jeu de données. Avec notre approche de relaxation, il est alors possible de trouver des solutions plus proches des valeurs des seuils. Comme attendu, plus λ est grand, plus le nombre de règles inattendues augmente. En effet, quand λ augmente, les fréquences des contraintes relaxées diminuent, par conséquent le nombre de règles inattendues augmente.

Comparaison avec le travail précédent

Les règles inattendues sont un cas particulier des règles rare. Mais à la différence des règles d'exception, où on cherche une règle qui contredit une règle générale, dans les règles inattendues on cherche une règle qui contredit une croyance $U \rightarrow V$, c'est à dire une règle donnée par l'utilisateur (avec un paramètre γ_{belief}).

D'après le travail effectué par M. Khiari [17], même si la modélisation CSP des règles inattendues est faite il est difficile de trouver des solutions, même pour des jeux de données petits. D'où l'intérêt d'utiliser la relaxation de contraintes sur ce travail. Pour trouver des solutions "*proches*" et pouvoir en tirer de conclusions de ces solutions.

Chapitre 7

Découverte de fragments toxicophores

7.1 Présentation de l'application

Dans ce chapitre, nous proposons un schéma de relaxation montrant l'intérêt de notre approche sur une application réelle dans le domaine de la chémoinformatique. Ce travail s'inscrit dans une collaboration avec le CERMN (UPRES EA 4258, Université de Caen Basse-Normandie).

La toxicologie est la science étudiant les substances chimiques toxiques. Elle s'intéresse notamment à l'identification de fragments moléculaires spécifiques au sein de la structure d'une molécule appelés toxicophores et considéré comme responsable direct des propriétés toxiques d'une substance chimique. Un objectif majeur est alors la découverte de tels fragments afin de mieux identifier les caractéristiques des molécules liées à la toxicité. Un motif chimique émergent est une conjonction de descripteurs moléculaires qui apparaît fréquemment dans une classe de molécules et peu fréquemment dans une autre classe [4, 3]. Ces motifs sont définis par les mesures de fréquence et de taux de croissance explicitées à la section 7.2.

Récemment, G. Poezevara a proposé une méthode fondée sur l'enchaînement d'une technique de recherche de sous-graphes fréquents utilisée pour changer la description des données avec une méthode récente de fouille sous contraintes dans le cas de données binaires afin d'extraire l'ensemble correct et complet de motifs émergents de graphes [29]. Les contraintes de fréquence et d'émergence (cette dernière étant issue de la mesure de taux de croissance) définissent ces motifs qui s'avérent précieux pour la prédiction de la toxicité [33].

Il est naturel de souhaiter combiner ces contraintes caractérisant les motifs du point de vue de leur présence dans les données avec des connaissances chimiques comme *l'aromaticité* ou *la densité* ou d'une molécule qui sont des indicateurs connus de la toxicité. Dans ce chapitre, les molécules sont décrites par des attributs correspondant à des sous-graphes initialement extraits des molécules et pour lesquels il est possible d'attacher des valeurs de propriétés chimiques, comme l'aromaticité ou la densité. Pour effectuer le processus de fouille, il est alors nécessaire de déclarer une requête combinant 4 mesures. Si il n'est pas toujours simple de fixer un seuil pour ces mesures, il est encore plus délicat d'en fixer 4 simultanément. En permettant de relâcher certaines de ces propriétés, nous pensons que notre cadre de relaxation facilite ainsi leur optimisation simultanée. De plus, il est simple de privilégier certaines propriétés : typiquement, les propriétés chimiques

sont ici vues comme prépondérantes.

Nous avons utilisé un jeu de données de 564 molécules (base ECB [1] fournis par le CERMN. La toxicité est une valeur continue mais celle-ci est classiquement divisée en 4 classes : classe $R50$ (molécules très toxiques), $R51$ (molécules moyennement toxiques), $R52$ (molécules faiblement toxiques) et 007 (molécules non toxiques). Le jeu de données utilisé est partitionnée en deux sous-ensembles : l'ensemble $\mathcal{D}_1$ contient des molécules très toxiques (372 molécules, classe R50) et l'ensemble $\mathcal{D}_2$ contient des molécules non toxiques (192 molécules, classe R52).

7.2 Requêtes pertinentes pour la découverte de fragments toxicophores

Dans ce travail, on s'intéresse à la découverte de toxicophores des molécules de la classe R50 versus R52. Une difficulté majeure de la tâche est le nombre potentiels de motifs qui est très grand. Il devient alors important de réduire le nombre de motifs extraits à ceux présentant un intérêt potentiel exprimé par l'utilisateur sous forme de contraintes. Ci-dessous une description des différentes contraintes que nous avons retenues :

- **Contrainte d'émergence** : les motifs émergents sont des motifs dont la fréquence varie fortement entre deux classes données. Cette contrainte, qui repose sur le taux de croissance, permet de caractériser une molécule d'une classe (toxique) par rapport à une autre classe (non-toxique) :

$$q_1(X) \equiv \frac{|\mathcal{D}_2| \times freq(X,\mathcal{D}_1)}{|\mathcal{D}_1| \times freq(X,\mathcal{D}_2)} \geq min\rho$$

où $min\rho$ est un seuil minimal pour le taux de croissance.

- **Contrainte de fréquence** : les motifs avec une fréquence très faible sont souvent dûs à des artefacts dans les données et sont du bruit. Pour les éliminer, on ajoute à la contrainte précédente, une contrainte de fréquence :

$$q_2(X) \equiv q_1(X) \wedge freq(X) \geq minfr$$

où $minfr$ est un seuil minimal pour la fréquence.

- **Contrainte d'aromacité** : pour chaque sous-graphe (donc chaque attribut) est associé une valeur de l'aromaticité qui est une mesure chimique. L'intérêt de cette mesure est qu'elle traduit une hypothèse toxicophore : plus un attribut a une forte valeur d'aromaticité, plus une molécule supportant cet attribut tend à être toxique. l'aromaticité d'un motif est la moyenne de l'aromaticité de ses attributs. Soit m une mesure donnant l'aromaticité d'un attribut. La requête q_3 permet alors d'extraire des motifs intégrant en plus une connaissance chimique portant sur l'aromaticité de ces attributs :

$$q_3(X) \equiv q_2(X) \wedge m(X) \geq min_{aromaticite}$$

où $min_{aromaticite}$ est un seuil minimal pour l'aromaticité.

- **Contrainte de densité** : la requête précédente peut être complétée en ajoutant une seconde connaissance chimique sur la densité d'un attribut (codant les molécules). Plus un sous-graphe est dense, plus ce sous-graphe est "solide" ; un motif composé d'attributs

1. European Chemicals Bureau.

denses renforce l'hypothèse d'un toxicophore. Il est donc judicieux de combiner la densité avec la mesure d'aromacité. On obtient alors la requête q_4 suivante :

$$q_4(X) \equiv q_3(X) \wedge d(X) \geq min_{densite}$$

où d est la mesure de densité et $min_{densite}$ le seuil minimal de densité.

Typiquement, la densité se calcule en prenant le nombre d'arêtes du sous-graphe divisé par le nombre de sommets du graphe au carré, c'est-à-dire : $d(X) = \frac{nbA(X)}{nbS(X)2}$ où nbA est le nombre d'arêtes et nbS le nombre de sommets.

Comme on peut le voir, nous avons plusieurs mesures dont il est difficile de donner les seuils pour obtenir les contraintes correspondantes et il est encore plus difficile de donner un seuil approprié pour une mesure par rapport aux autres seuils donnés. D'où l'intérêt de relâcher les seuils, ce qui donne, de façon relative, moins d'importance au choix des seuils.

Enfin, en se qui concerne les sémantiques de violation, on peut a priori faire plus confiance à la connaissance du domaine (notamment l'aromaticité) et donc pénaliser plus fortement une violation de la contrainte liée à l'aromaticité que les autres contraintes.

7.3 Transformation

Pour cette requête, on a décidé d'utiliser la sémantique de violation μ_2(c'est à dire l'écart relatif), parce-que l'on va relaxer 4 contraintes de nature hétérogène, d'où l'idée on normaliser les écarts. On fait les transformations comme suit :

$$\rightarrow c_1 \equiv \frac{|\mathcal{D}_2|\times freq(X,\mathcal{D}_1)}{|\mathcal{D}_1|\times freq(X,\mathcal{D}_2)} \geq min\rho \quad \rightarrow \mu_2(X) = \begin{cases} 0 & si \ \frac{|\mathcal{D}_2|\times freq(X,\mathcal{D}_1)}{|\mathcal{D}_1|\times freq(X,\mathcal{D}_2)} \geq min\rho \\ \frac{min\rho - \frac{|\mathcal{D}_2|\times freq(X,\mathcal{D}_1)}{|\mathcal{D}_1|\times freq(X,\mathcal{D}_2)}}{min\rho} & sinon \end{cases}$$

$$\rightarrow c_2 \equiv freq(X) \geq minfr \qquad \rightarrow \mu_2(X) = \begin{cases} 0 & si \ freq(X) \geq minfr \\ \frac{minfr - freq(X)}{minfr} & sinon \end{cases}$$

$$\rightarrow c_3 \equiv m(X) \geq min_{aromaticite} \quad \rightarrow \mu_2(X) = \begin{cases} 0 & si \ m(X) \geq min_{aromaticite} \\ \frac{min_{aromaticite} - m(X)}{min_{aromaticite}} & sinon \end{cases}$$

$$\rightarrow c_4 \equiv d(X) \geq min_{densite} \qquad \rightarrow \mu_2(X) = \begin{cases} 0 & si \ d(X) \geq min_{densite} \\ \frac{min_{densite} - d(X)}{min_{densite}} & sinon \end{cases}$$

Maintenant on va définir la formulation pour la modélisation sous forme d'un CSP.

Formulation

La table 7.1 décrit l'ensemble des contraintes primitives modélisant la requête proposée pour la découverte de toxicophores.

— La variable ensembliste $\{X\}$ représente le motif recherché.
— Les variables entières $\{E,F,M,D\}$ représentent l'émergence, la fréquence, la aromaticité et la densité de X respectivement.

Contrainte	Formulation
$\frac{\lvert\mathcal{D}_2\rvert \times freq(X,\mathcal{D}_1)}{\lvert\mathcal{D}_1\rvert \times freq(X,\mathcal{D}_2)} \geq min\rho$	$E \geq min\rho$
$freq(X) \geq minfr$	$F \geq minfr$
$m(X) \geq min_{aromaticite}$	$M \geq min_{aromaticite}$
$d(X) \geq min_{densite}$	$D \geq min_{densite}$

TABLE 7.1 – Formulation des contraintes pour la découverte de toxicophores

— Contraintes numériques : $\mathcal{C} = \{(E \geq min\rho), (F \geq minfr), (M \geq min_{aromaticite}), (D \geq min_{densite})$

Modélisation sous forme d'un CSP

Maintenant on définit le CSP $\mathcal{P}$=$(\mathcal{X}, \mathcal{D}, \mathcal{C})$ où :

a) $\mathcal{X} = \{X\} \cup \{X_{ent}\} \cup \{z_1, z_2, z_3, z_4, Z\}$ est l'ensemble des variables :
   — La variable ensembliste $\{X\}$ représente le motif recherché.
   — Les variables entières X_{ent}=$\{E,F,M,D\}$ représentent l'émergence, la fréquence, la aromaticité et la densité de X respectivement.
   — Les variables de coût $\{z_1,z_2,z_3,z_4\}$ qui quantifient la violation pour les 4 contraintes de la requête et la variable de coût globale Z pour quantifier la violation globale.

b) $\mathcal{D} = D_X \cup D_E \cup D_F \cup D_M \cup D_D \cup \{D_{z_1}, D_{z_2}, D_{z_3}, D_{z_4}, D_Z\}$ est l'ensemble des domaines :
   — Le domaine de la variable ensembliste D_X =$\{\emptyset, \ldots, \mathcal{I}\}$
   — Les domaines des variables entières :
      — D_E =$\{0,\ldots,1\}$.
      — D_F =$\{0,\ldots,m\}$ où m=nombre de transactions.
      — D_M =$\{0,\ldots,\}$
      — D_D =$\{0,\ldots,1\}$
   — Comme on utilise la sémantique de violation μ_2 (écart relatif), alors les valeurs des variables z_1, z_2, z_3 et z_4 sont normalisées, donc les domaines des variables de coût $D_{z_1} = D_{z_2} = D_{z_3} = D_{z_4}$ =$\{0,\ldots, 1\}$ et D_Z =$\{0,\ldots,\lambda\}$ [2].

c) $\mathcal{C} = \mathcal{C}'_{num} \cup \{Z = \sum z_i\} \cup \{Z \leq \lambda\}$ est l'ensemble des contraintes :
   — Contraintes numériques :$\mathcal{C}_{num} = \{(E \geq min\rho), (F \geq minfr), (M \geq min_{aromaticite}), (d(X) \geq min_{densite})$ dont la relaxation $\mathcal{C}'_{num} = \{(z_1 = max(0, \frac{min\rho - E}{min\rho})), (z_2 = max(0, \frac{minfr - F}{minfr})), (z_3 = max(0, \frac{min_{aromaticite} - M}{min_{aromaticite}}))\}, (z_4 = max(0, \frac{min_{densite} - D}{min_{densite}}))$

2. λ :seuil de violation maximale

Conclusion et perspectives

Conclusion

Ce mémoire propose une solution au problème de la relaxation de contraintes pour l'extraction de motifs. La solution proposée, basée sur l'utilisation du cadre disjonctif, semble au vu des tests effectués mener à une extraction de motifs équivalente a l'ensemble de solutions, ne nécessite donc pas des *postprocessing*. La complexité des calculs est de plus très faible, rendant l'implémentation de cette méthode parfaitement réalisable.

Il reste cependant beaucoup à faire, notamment dans la prise en compte de préférences ou la possibilité de traiter d'autres types de contraintes n-aires. En effet, grâce à notre cadre de relaxation, nous avons proposé la relaxation pour deux contraintes n-aires très connues (règles d'exception, règles inattendues). Cependant, dans les applications réelles, il existent d'autres types de contraintes n-aires qu'il est nécessaire de prendre en compte. Nous souhaitons étendre notre cadre de relaxation pour d'autres cas d'études comme le *clustering* où la relaxation de contraintes n-aires est souvent nécessaire pour trouver des solutions.

Perspectives

i) Relaxation de contraintes ensemblistes : définir les sémantiques de violation et les transformations nécessaires pour faire la relaxation des contraintes ensemblistes.

   Comme exemple on peut définir quelques sémantiques de violation pour certaines contraintes ensemblistes :
   — $X_p = X_q \rightarrow z_{=} = card(X_p \setminus X_q) + card(X_q \setminus X_p)$
   — $X_p \subseteq X_q \rightarrow z_{\subseteq} = card(X_p \setminus X_q)$

ii) Traiter d'autres contraintes n-aires : Ils existent plusieurs types de contraintes n-aires que l'on aimerait relaxer, notamment *le clustering* qui est très utilisé dans plusieurs domaines d'application, où il est parfois très difficile de satisfaire toutes les contraintes du problème.

iii) Prise en compte des préférences : dans certains cas l'utilisateur aimerait exprimer des préférences entre les contraintes et/ou les solutions.

Bibliographie

[1] R. Agrawal, T. Imielinski, and A. N. Swami. Mining association rules between sets of items in large databases. *SIGMOD Conference*, pages 207–216, 1993.

[2] C. Antunes and A. L. Oliveira. Constraint relaxations for discovering unknown sequential patterns. *Knowledge Discovery in Inductive Databases, Proceedings of the Third International Workshop on Knowledge Discovery in Inductive Databases*, 3377 :11–32, 2004.

[3] Jurgen Bajorath. Simulation of Sequential Screening Experiments Using Emerging Chemical Patterns. *Medicinal Chemistry*, 4 :80–90, 2008.

[4] Jurgen Bajorath and Jens Auer. Emerging Chemical Patterns : A New Methodology for Molecular Classification and Compound Selection. *Journal of Chemical Information and Modeling*, 46 :2502–2514, 2006.

[5] R. J. Bayardo. The hows, whys, and whens of constraints in itemset and rule discovery. *In Proc. of the Workshop on Inductive Databases and Constraint Based Mining (IDW'05)*, pages 207–216, 2005.

[6] Christian Bessière and Marie-Odile Cordier. Arc-consistency and arc-consistency again. In *AAAI*, pages 108–113, 1993.

[7] Christian Bessière, Eugene C. Freuder, and Jean-Charles Régin. Using constraint metaknowledge to reduce arc consistency computation. *Artif. Intell.*, 107(1) :125–148, 1999.

[8] Christian Bessière and Jean-Charles Régin. Refining the basic constraint propagation algorithm. In Bernhard Nebel, editor, *IJCAI*, pages 309–315. Morgan Kaufmann, 2001.

[9] Stefano Bistarelli and Francesco Bonchi. Interestingness is not a dichotomy : Introducing softness in constrained pattern mining. In *Knowledge Discovery in Databases (PKDD'05)*, volume 3721 of *LNCS*, pages 22–33. Springer, 2005.

[10] Stefano Bistarelli, Ugo Montanari, Francesca Rossi, Thomas Schiex, Gérard Verfaillie, and Hélène Fargier. Semiring-based csps and valued csps : Frameworks, properties, and comparison. *Constraints*, pages 199–240, 1999.

[11] Alan Borning, Bjorn N. Freeman-Benson, and Molly Wilson. Constraint hierarchies. *Lisp and Symbolic Computation*, 16(8) :223–270, 1992.

[12] Stephan Borzsonyi, Donald Kossmann, and Konrad Stocker. The skyline operator. In *Proceedings of the 17th International Conference on Data Engineering(ICDE'01)*, IEEE Computer Science, pages 421–430. Springer, 2001.

[13] Romuald Debruyne. *Etude des consistances locales pour les problmes de satisfaction de contraintes de grande taille.* PhD thesis, LIRMM-Université de Montpellier II, Dec 1998.

[14] Eugene C. Freuder and Richard J. Wallace. Partial constraint satisfaction. *Artificial Intelligence*, 16(8) :21–70, 1992.

[15] M. N. Garofalakis, R. Rastogi, and K. Shim. SPIRIT : Sequential pattern mining with regular expression constraints. *In The VLDB Journal*, pages 223–234, 1999.

[16] Gecode. Gecode : Generic constraint development environment, 2006. Available from `http://www.gecode.org`.

[17] M. Khiari, P. Boizumault, and B. Crémilleux. Constraint programming for mining n-ary patterns. In *16th International Conference on Principles and Practice of Constraint Programming (CP'10)*, volume 6308 of *LNCS*, pages 552–567, St Andrews, Scotland, 2010. Springer.

[18] M. Khiari, P. Boizumault, and B. Crémilleux. Combining CSP and constraint-based mining for pattern discovery. In *Advances in Knowledge Discovery and Management (Post-EGC Selected Papers)*. Springer, 2011. 20 pages.

[19] Alan K. Mackworth. Consistency in networks of relations. *Artif. Intell.*, 8(1) :99–118, 1977.

[20] Roger Mohr and Thomas C. Henderson. Arc and path consistency revisited. *Artif. Intell.*, 28(2) :225–233, 1986.

[21] Ugo Montanari. Networks of constraints : Fundamental properties and applications to picture processing. *Inf. Sci.*, 7 :95–132, 1974.

[22] Siegfried Nijssen and Tias Guns. Integrating constraint programming and itemset mining. In *European Conference, ECML PKDD 2010*, volume 6322 of *LNCS*, pages 467–482. Springer, 2010.

[23] Balaji Padmananabhan and Alexander Tuzhilin. A belief-driven method for discovering unexpected patterns. In *Knowledge Discovery in Databases (KDD'98)*, pages 94–100, 1998.

[24] Fréderic Pennerath and Amedeo Napoli. The model of most informative patterns and its application to knowledge extraction from graph databases. In *Machine Learning and Knowledge Discovery in Databases European Conference(ECML/PKDD 2009)*, volume 5782, pages 205–220. Springer, 2009.

[25] Thierry Petit. Modélisation et algorithmes de résolution de problèmes sur-contraints. In *PhD thesis*. LIRMM - Université de Montpellier II, 2002.

[26] Thierry Petit, Jean-Charles Régin, and Christian Bessière. Meta-constraints on violations for over constrained problems. In *ICTAI*, pages 358–365. IEEE Computer Society, 2000.

[27] Thierry Petit, Jean-Charles Régin, and Christian Bessière. Specific filtering algorithms for over-constrained problems. In *7th International Conference on Principles and Practice of Constraint Programming (CP'01)*, volume 2239 of *LNCS*, pages 451–463. Springer, 2001.

[28] Guillaume Poezevara, Bertrand Cuissart, and Bruno Crémilleux. Discovering Emerging Graph Patterns from Chemicals. *ISMIS*, pages 45–55, 2009.

[29] Guillaume Poezevara, Bertrand Cuissart, and Bruno Crémilleux. Extracting and summarizing the frequent emerging graph patterns from a dataset of graphs. *Journal of Intelligent Systems(JIIS)*, pages 1–21, 2011.

[30] Guillaume Poezevara, Bertrand Cuissart, Bruno Crémilleux, and Ryan Bissel-Siders. Mining patterns and subgraphs as potential toxicophores to predict contextual ecotoxicity. *5th Workshop on Computers in Scientific Discovery*, 2010.

[31] Guillaume Poezevara, Bertrand Cuissart, Bruno Crémilleux, Sylvain Lozano, Marie-Pierre Halm-Lemeille, Elodie Lescot-Fontaine Alban Lepailleur, Ryan Bisell-Siders, Sylvain Rault, and Ronan Bureau. Supervised classification and QSAR in ecotoxicology : comparaison of two methods. *Journées nationales 2009 de la société française de chemoinformatique*, 2009.

[32] Guillaume Poezevara, Bertrand Cuissart, Bruno Crémilleux, Sylvain Lozano, Marie-Pierre Halm-Lemeille, Elodie Lescot-Fontaine Alban Lepailleur, Ryan Bisell-Siders, Sylvain Rault, and Ronan Bureau. Assessment of chemical risk phrases in ecotoxicology : comparaison of two methods. *4th International Symposium of Toxicity Assessment*, 2010.

[33] Guillaume Poezevara, Bertrand Cuissart, Bruno Crémilleux, Sylvain Lozano, Marie-Pierre Halm-Lemeille, Elodie Lescot-Fontaine Alban Lepailleur, Ryan Bisell-Siders, Sylvain Rault, and Ronan Bureau. Introduction of Jumping Fragments in Combination with QSARs for the Assessment of Classification in Ecotoxicology. *Journal of Chemical Information and Modeling(JCIM)*, pages 1330–1339, 2010.

[34] Luc De Raedt, Tias Guns, and Siegfried Nijssen. Constraint programming for itemset mining. In *14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'08)*, pages 204–212. ACM, 2008.

[35] Luc De Raedt, Tias Guns, and Siegfried Nijssen. Correlated itemset mining in roc space :a constraint programming approach. In *International Conference on Knowledge Discovery and Data Mining (KDD'09)*, pages 647–655, 2009.

[36] Luc De Raedt, Tias Guns, and Siegfried Nijssen. Constraint programming for data mining and machine learning. In *Twenty-Fourth AAAI Conference on Artificial Inteligence(AIII-10)*, pages 1671–1675, 2010.

[37] F. Rossi, P. Van Beek, and T. Walsh. Handbook of Constraint Programming(Foundations of Artificial Intelligence). *Elvier Science*, 2006.

[38] Thomas Schiex, Hélène Fargier, and Gérard Verfaillie. Valued constraint satisfaction problems : Hard and easy problems. *In IJCAI (1)*, pages 631–639, 1995.

[39] A. Soulet and B. Crémilleux. Optimizing constraintbased mining by automatically relaxing constraints. *In Proceedings of the 5th IEEE International Conference on Data Mining (ICDM 2005)*, page 777, 2005.

[40] E. Suzuki. Undirected Discovery of Interesting Exception Rules. *Int. Journal of Pattern Recognition and Artificial Intelligence*, 16(8) :1065–1086, 2002.

[41] L. Szathmary, A. Napoli, and P. Valtchev. Towards rare itemset mining. In *Proceedings of the 19th IEEE ICTAI '07*, volume 1, 2007.

[42] David Waltz. Understanding line drawings of scenes with shadows. In *The Psychology of Computer Vision*, pages 19–91. McGraw-Hill, 1975.

[43] K. Wang, Y.Jiang, J. X. Yu, G. Dong, and J. Han. Divide-andapproximate : A novel constraint push strategy for iceberg cube mining. *IEEE Trans. Knowl. Data Eng.*, pages 354–368, 2005.